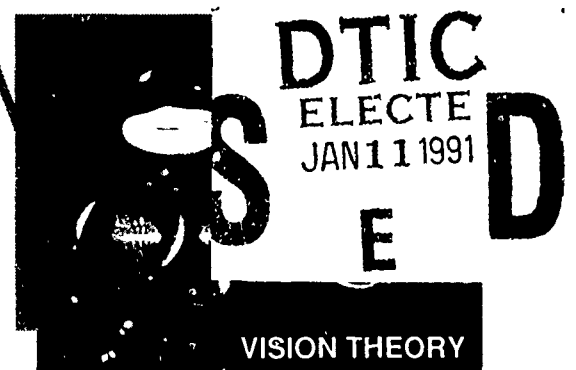
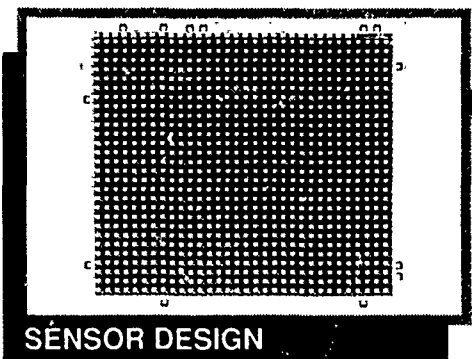
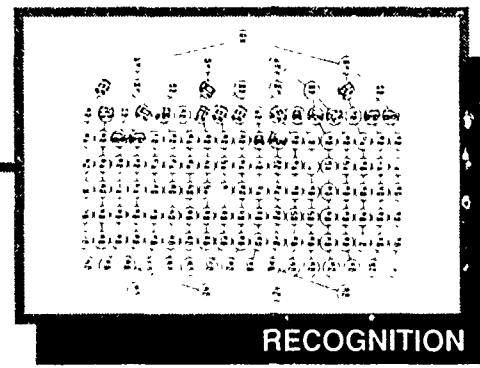
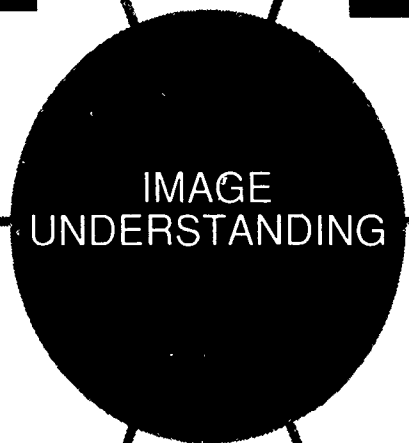
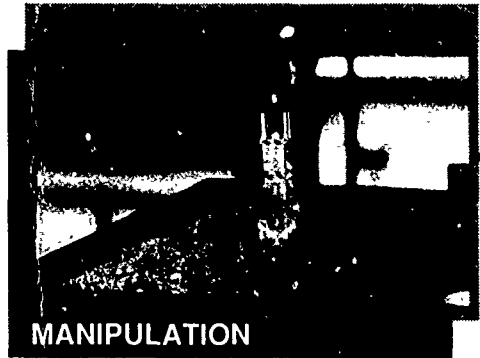
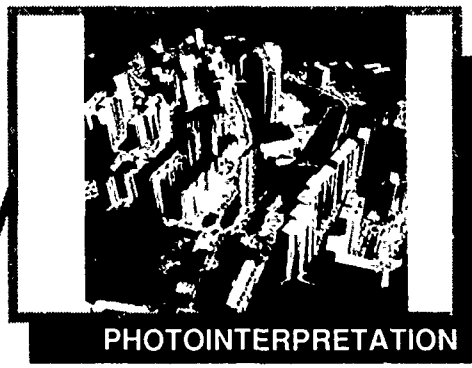


2

PROCEEDINGS:

Image Understanding Workshop



AD-A231 063



Sponsored by:
Defense Advanced Research Projects Agency
Information Science and Technology Office



September 1990

DISTRIBUTION STATEMENT A
Approved for public release
Distribution Unlimited

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Proceedings: Image Understanding Workshop September 1990		5. TYPE OF REPORT & PERIOD COVERED ANNUAL TECHNICAL June 1989-September 1990
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Lee S. Baumann (Editor)		8. CONTRACT OR GRANT NUMBER(s) N00014-86-C-0700
9. PERFORMING ORGANIZATION NAME AND ADDRESS Science Applications International Corp. 1710 Goodridge Drive McLean, VA 22102		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ARPA Order 5605
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Blvd. Arlington, VA 22209		12. REPORT DATE September 1990
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		13. NUMBER OF PAGES 921
		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) APPROVED FOR PUBLIC RELEASE, DISTRIBUTION UNLIMITED		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Computer Vision; Digital Image Processing; Image Understanding; Scene Analysis; Automatic Target Recognition; Robotic Vision; Image Segmentation; Edge Detection; I.U. Systems Environment.		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This document contains the annual progress reports and technical pa- pers presented by the research activities in Image Understanding sponsored by the Information Science & Technology Office, Defense Advanced Research Projects Agency. The papers were presented at a workshop held on 11-13 September 1990, in Pittsburgh, Pennsylvania. Also included are additional technical papers from the research ac- tivities which were not presented due to lack of time but are ger- mane to this research field.		



Image Understanding Workshop

Proceedings of a Workshop
Held at
Pittsburgh, Pennsylvania

September 11-13, 1990

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

Sponsored by:
Defense Advanced Research Projects Agency
Information Science and Technology Office

DTIC
ELECTE
JAN 11 1991
S E D

This document contains copies of reports prepared for the DARPA Image Understanding Workshop. Included are Principal Investigator reports and technical results from both the basic and strategic computing programs within DARPA/ISTO sponsored projects.

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States Government.

Distributed by

Morgan Kaufmann Publishers, Inc.

2929 Campus Drive

San Mateo, CA 94403

ISBN 1-55860-140-6

Printed in the United States of America

TABLE OF CONTENTS

	<u>PAGE</u>
AUTHOR INDEX.....	xiii
ACKNOWLEDGMENTS.....	xvii
SECTION I: PRINCIPAL INVESTIGATORS REPORTS	
"CMU Image Understanding Program", Takeo Kanade and Steven Shafer; Carnegie Mellon University.....	1
"Image Understanding and Robotics Research at Columbia University", John R. Kender, Peter K. Allen, and Terrance E. Boulton; Columbia University.....	11
"MIT Progress in Understanding Images", T. Poggio and the staff; Massachusetts Institute of Technology.....	19
"Maryland Progress in Image Understanding", John (Yiannis) Aloimonos, Larry S. Davis, and Azriel Rosenfeld; University of Maryland.....	28
"USC Image Understanding Research: 1989-1990", R. Nevatia, K. Price, and G. Medioni; University of Southern California.....	43
"Image Understanding Research at GE", J.L. Mundy; GE Corporate Research and Development Center.....	67
"Image Understanding Research at Honeywell", Bir Bhanu; Honeywell Systems and Research Center.....	70
"Image Understanding Research at Rochester", Christopher Brown and Randal Nelson; University of Rochester.....	76
"Progress in Computer Vision at the University of Massachusetts", Edward M. Riseman and Allen R. Hanson; University of Massachusetts at Amherst.....	86
"Image Understanding Research at SRI International", Martin A. Fischler and Robert C. Bolles; SRI International.....	97
"Image Understanding at the GRASP Laboratory", Ruzena Bajcsy; University of Pennsylvania.....	107

	<u>PAGE</u>
"Progress Toward an Image Understanding Application Development Environment", Tod S. Levitt, Scott E. Johnston, Scott Barclay, and John W. Dye; Advanced Decision Systems, Daryl T. Lawton; Georgia Institute of Technology.....	115
"Image Understanding Research at Brown University", D.B. Cooper, T.L. Dean, and W.A. Wolovich; Brown University.....	131
"IU at UI: An Overview of Research During 1988-90", Narendra Ahuja; University of Illinois.....	134
"Vision-Based Navigation", William B. Thompson; University of Minnesota.....	141

SECTION II: TECHNICAL PAPERS

Physics-Based Low Level Vision

"New Results in Shape from Shading", J. Oliensis; University of Massachusetts at Amherst.....	145
"The Mathematical Foundations of Smoothness Constraints: A New Class of Coupled Constraints", M.A. Snyder; University of Massachusetts at Amherst...	154
"A Photometric Invariant and Shape Constraints at Parabolic Points", Lawrence B. Wolff; Columbia University.....	162
"Image Blurring Effects Due to Depth Discontinuities", Thang C. Nguyen and Thomas S. Huang; University of Illinois.....	174
"Illuminant Precompensation for Texture Discrimination Using Filters", Robert S. Thau; Massachusetts Institute of Technology.....	179
"Surface Reflection: Physical and Geometrical Perspectives", Shree K. Nayar, Katsushi Ikeuchi, and Takeo Kanade; Carnegie Mellon University.....	185
"Local Spatial Frequency Analysis for Computer Vision", John Krumm and Steven A. Shafer; Carnegie Mellon University.....	213
"Least Median of Squares Based Robust Analysis of Image Structure", Peter Meer, Doron Mintz, and Azriel Rosenfeld; University of Maryland.....	231

	<u>PAGE</u>
"A Fast, High Breakdown Point Robust Estimator for Computer Vision Applications", Doron Mintz, Peter Meer, and Azriel Rosenfeld; University of Maryland.....	255
<u>Motion and Stereo</u>	
"Shape and Motion without Depth", Carlo Tomasi and Takeo Kanade; Carnegie Mellon University.....	258
"A Unified Theory of Structure from Motion", Minas E. Spetsakis and John (Yiannis) Aloimonos; University of Maryland.....	271
"FIXATION: A Direct Method for Recovery of Motion and Shape in the General Case", M. Ali Taalebi-Nezhaad; Massachusetts Institute of Technology.....	284
"Direct Non-Linear Methods for Recovering Structure and Motion", David J. Michael; Massachusetts Institute of Technology.....	292
"Robustness of Correspondence-Based Structure from Motion", R. Dutta and M.A. Snyder; University of Massachusetts at Amherst.....	299
"Comparative Results of Some Motion Algorithms on Real Image Sequences", Harpreet S. Sawhney and Allen R. Hanson; University of Massachusetts at Amherst.....	307
"An Estimation - Theoretic Framework for Image-Flow Computation", Ajit Singh; Columbia University.....	314
"Qualitative Detection of Motion by a Moving Observer", Randal C. Nelson; University of Rochester.....	329
"Multiple Frame Analysis of Translation Dominant Motion", Yong Cheol Kim and Keith Price; University of Southern California.....	339
"Detecting Moving Objects from a Moving Platform", J. Frazier and R. Nevatia; University of Southern California.....	348
"Motion and Binocular Stereo Integrated System for Passive Ranging", Peter F. Symosek, Bir Bhanu, Scott Snyder, and Barry Roberts; Honeywell Systems and Research Center.....	356

	<u>PAGE</u>
"Inertial Navigation Sensor Integrated Motion Analysis for Autonomous Vehicle Navigation", Barry Roberts and Bir Bhanu; Honeywell Systems and Research Center.....	364
"Temporal Integration of Visual Surface Reconstruction", Joachim Heel and Satyajit Rao; Massachusetts Institute of Technology.....	376
"A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment", Takeo Kanade and Masatoshi Okutomi; Carnegie Mellon University.....	383
"General Model-Based 3D Surface Estimation, Recognition and Segmentation from Multiple Images", David B. Cooper, Yi-Ping Hung, Jayashree Subrahmonia; Brown University.....	399
"FLAT MMF: A New Recursive Technique for the 3-D Motion Analysis", Siu-Leong Iu and Kwangyoen Wahn; University of Pennsylvania.....	405
"Integrated Multiresolution Image Acquisition and Surface Reconstruction from Active Stereo", Subhdev Das and Narendra Ahuja; University of Illinois.....	418
"A Dynamical Systems Approach to Integration in Stereo", Edward J. Altman and Narendra Ahuja; University of Illinois.....	423
"Experimental Results of 3D Motion Estimation Using Images of Outdoor Scenes", M.K. Leung, Y.C. Liu, and T.S. Huang; University of Illinois.....	428
"Deriving Line and Surface Orientation by Statistical Methods", Robert T. Collins and Richard S. Weiss; University of Massachusetts at Amherst.....	433
"Computational and Biological Models of Stereo Vision", Stephen T. Barnard and Martin A. Fischler; SRI International.....	439
"Recent Progress in CYCLOPS: A System for Stereo Cartography", Stephen T. Barnard, SRI International.....	449

Object Recognition and Scene Analysis

"A Context-Based Recognition System for Natural Scenes and Complex Domains", Thomas M. Strat and Martin A. Fischler; SRI International.....	456
"Region Grouping Using the Minimum-Description-Length Principle", Yvan G. Leclerc; SRI International.....	473
"How to Decide from the First View Where to Look Next", Jasna Maver and Ruzena Bajcsy; University of Pennsylvania.....	482
"Task Oriented Vision", Katsushi Ikeuchi and Martial Hebert; Carnegie Mellon University.....	497
"Recognition and Positioning of Piecewise Algebraic Objects", Gabriel Taubin and David B. Cooper; Brown University.....	508
"Modeling Polyhedra by Constraints", Van-Duc Nguyen and Joseph L. Mundy; GE Corporate Research and Development Center, Deepak Kapur; State University of New York.....	515
"MARVEL: Location Recognition Using Stereo Vision", David J. Braunegg; Massachusetts Institute of Technology.....	530
"TOSS - A System for Efficient Three Dimensional Object Recognition", Fridtjof Stein and Gerard Medioni; University of Southern California.....	537
"Recovering Shape from Contour for SHGCs and CGCs", Fatih Ulupinar and Ramakant Nevatia; University of Southern California.....	544
"Recovery of Generalized Cylinders from a Single Intensity View", Ari D. Gross and Terrance E. Boulton; Columbia University.....	557
"Energy-Based Segmentation of Very Sparse Range Surfaces", Terrance E. Boulton and Mark Lerner; Columbia University.....	565
"Straight Homogeneous Generalized Cylinders: Constraints from Contour", Ari D. Gross; Columbia University.....	573

	<u>PAGE</u>
"Self-Optimizing Control System for Adaptive Image Segmentation", Bir Bhanu, Sungkee Lee, and John Ming; Honeywell Systems and Research Center.....	583
"Extensions of a Theory of Networks for Approximation and Learning: Dimensionality Reduction and Clustering", Tomaso Poggio and Federico Girosi; Massachusetts Institute of Technology.....	597
"Model-Group Indexing for Recognition", David Clemens and David Jacobs; Massachusetts Institute of Technology.....	604
"The Skeleton Sketch: Finding Salient Frames of Reference", J. Brian Subirana-Vilanova; Massachusetts Institute of Technology.....	614
"Indexing Via Color Histograms", Michael J. Swain and Dana H. Ballard; University of Rochester.....	623
"Real-time Qualitative Detection of Multi-colored Objects for Object Search", Lambert E. Wixson; University of Rochester.....	631
"Perceptual Organization of Occluding Contours", Lance R. Williams; University of Massachusetts at Amherst.....	639
"View Variation of Point Set and Line Segment Features", J. Brian Burns, Richard Weiss, and Edward M. Riseman; University of Massachusetts at Amherst.....	650
"Pose Refinement: Application to Model Extension and Sensitivity to Camera Parameters", Rakesh Kumar and Allen R. Hanson; University of Massachusetts at Amherst.....	660

Aerial Photointerpretation

"Recovering 3D Information from Complex Aerial Imagery", Yuan C. Hsieh, Frederic Perlant, and David M. McKeown; Carnegie Mellon University.....	670
"Fusion of Monocular Cues to Detect Man-Made Structures in Aerial Imagery", Jeffrey Shufelt and David M. McKeown; Carnegie Mellon University.....	692

	<u>PAGE</u>
"Map-Based Localization: The 'Drop-Off' Problem", William B. Thompson, Herbert L. Pick, Jr., Bonnie H. Bennett, Marian R. Heinrichs, Steven L. Savitt and Kip Smith; University of Minnesota.....	706
B-snakes: Implementation and Application to Stereo", Sylvie Menet, Philippe Saint-Marc, and Gerard Medioni; University of Southern California.....	720
"Benchmark Evaluation of a Model-Based Object Recognition System", A.J. Heller and J.L. Mundy; GE Corporate Research and Development Center.....	727
"A Multistrategy Learning Approach for Target Model Recognition, Acquisition, and Refinement", John Ming and Bir Bhanu; Honeywell Systems and Research Center.....	742
"A Formalization and Implementation of Topological Visual Navigation in Two Dimensions", John R. Kender, Il-Pyung Park, and David Yang; Columbia University.....	757
"Annotated Maps for Autonomous Land Vehicles", Charles Thorpe and Jay Gowdy; Carnegie Mellon University.....	765
"Experiments in Autonomous Navigation", Claude Fennema and Allen R. Hanson; University of Massachusetts at Amherst.....	772
"Image-Based Navigation Using 360° Views", Jia-Wei Hong and Xiaonan Tan; New York University, Brian Pinette, Richard Weiss, and Edward M. Riseman; University of Massachusetts at Amherst.....	782
"Reactive and Preplanned Control in a Mobile Robot", Monnett Hanvey Soldo; Columbia University.....	792

IU Systems Environment and Vision Systems

"General Routing on the Lowest Level of the Image Understanding Architecture", Martin C. Herbordt and Charles C. Weems; University of Massachusetts at Amherst, David B. Shu; Hughes Research Laboratories.....	797
---	-----

PAGE

"A Parallel Algorithm for List Ranking Image Curves in $O(\log N)$ Time", Ling Tony Chen and Larry S. Davis; University of Maryland.....	805
"Purposive and Qualitative Active Vision", John (Yiannis) Aloimonos; University of Maryland.....	816
"Efficient Parallel Processing in High Level Vision", Craig Reinhart and Ramakant Nevatia; University of Southern California.....	829
"Selective Attention as Sequential Behavior: Modeling Eye Movements with an Augmented Hidden Markov Model", Raymond D. Rimey and Christopher M. Brown; University of Rochester.....	840
"Satisfying the Resolution Constraint in the 'MVP' Machine Vision Planning System", Konstantinos Tarabanis and Peter K. Allen; Columbia University, Roger Y. Tsai; IBM T. J. Watson Research Center.....	850
"Statistical Decision Theory for Sensor Fusion", Raymond McKendall; University of Pennsylvania.....	861
"Robust Multi-Sensor Fusion: A Decision - Theoretic Approach", Gerda Kamberova and Max Mintz; University of Pennsylvania.....	867
"Non-Monotonic Decision Rules for Sensor Fusion", Raymond McKendall and Max Mintz; University of Pennsylvania.....	874
"Real-Time Vergence Control for Binocular Robots", Thomas J. Olson; University of Virginia, David J. Coombs; University of Rochester.....	881

Robotics Applications

"Sequential Decision Making for Active Perception", Thomas Dean, Theodore Camus, and Jak Kirman; Brown University.....	889
"Qualitative Visual Control of a Robot Manipulator", Jean-Yves Herve, Peter Cucka, Rajeev Sharma; University of Maryland.....	895

	<u>PAGE</u>
"Real-Time Visual Servoing", Peter Allen, Billibon Yoshimi, and Aleksandar Timcenko; Columbia University.....	909
"Force/Torque Sensing for Object Recognition", William A. Wolovich; Brown University.....	919

AUTHOR INDEX

Ahuja, N.	134, 418, 423
Allen, P. K.	11, 850, 909
Aloimonos, J. (Y.)	28, 271, 816
Altman, E. J.	423
Bajcsy, R.	107, 482
Ballard, D. H.	623
Barclay, S.	115
Barnard, S. T.	439, 449
Bennett, B. H.	706
Bhanu, B.	70, 356, 364, 583, 742
Bolles, R. C.	97
Boult, T. E.	11, 557, 565
Braunegg, D. J.	530
Brown, C. M.	76, 840
Burns, J. B.	650
Camus, T.	889
Chen, L. T.	805
Clemens, D.	604
Collins, R. T.	433
Coombs, D. J.	881
Cooper, D. B.	131, 399, 508
Cucka, P.	895
Das, S.	418
Davis, L. S.	28, 805
Dean, T. L.	131, 889
Dutta, R.	299
Dye, J. W.	115
Fennema, C.	772
Fischler, M. A.	97, 439, 456
Frazier, J.	348
Girosi, F.	597
Gowdy, J.	765
Gross, A. D.	557, 573
Hanson, A. R.	86, 307, 660, 772
Hebert, M.	497
Heel, J.	376
Heinrichs, M. R.	706
Heller, A. J.	727
Herbordt, M. C.	797
Herve, J.	895
Hong, J.	782
Hsieh, Y. C.	670
Huang, T. S.	174, 428
Hung, Y.	399

Ikeuchi, K.	185, 497
Iu, S.	405
Jacobs, D.	604
Johnston, S. E.	115
Kamberova, G.	867
Kanade, T.	1, 185, 258, 383
Kapur, D.	515
Kender, J. R.	11, 757
Kim, Y. C.	339
Kirman, J.	889
Krumm, J.	213
Kumar, R.	660
Lawton, D. T.	115
Leclerc, Y. G.	473
Lee, S.	583
Lerner, M.	565
Leung, M. K.	428
Levitt, T. S.	115
Liu, Y. C.	428
Maver, J.	482
McKendall, R.	861, 874
McKeown, D. M.	670, 692
Medioni, G.	43, 537, 720
Meer, P.	231, 255
Menet, S.	720
Michael, D. J.	292
Ming, J.	583, 742
Mintz, D.	231, 255
Mintz, M.	867, 874
Mundy, J. L.	67, 515, 727
Nayar, S. K.	185
Nelson, R. C.	76, 329
Nevatia, R.	43, 348, 544, 829
Nguyen, T. C.	174
Nguyen, V.	515
Okutomi, M.	383
Oliensis, J.	145
Olson, T. J.	881
Park, I.	757
Perlant, F.	670
Pick, H. L., Jr.	706
Pinette, B.	782
Poggio, T.	19, 597
Price, K.	43, 339

Rao, S.	376
Reinhart, C.	829
Rimey, R. D.	840
Riseman, E. M.	86, 650, 782
Roberts, B.	356, 364
Rosenfeld, A.	28, 231, 255
Saint-Marc, P.	720
Savitt, S. L.	706
Sawhney, H. S.	307
Shafer, S. A.	1, 213
Sharma, R.	895
Shu, D. B.	797
Shufelt, J.	692
Singh, A.	314
Smith, K.	706
Snyder, M. A.	154, 299
Snyder, S.	356
Soldo, M. H.	792
Spetsakis, M. E.	271
Stein, F.	537
Strat, T. M.	456
Subirana-Vilanova, J. B.	614
Subrahmonia, J.	399
Swain, M. J.	623
Symosek, P. F.	356
Taalebi-Nezhaad, M. A.	284
Tan, X.	782
Tarabanis, K.	850
Taubin, G.	508
Thau, R. S.	179
Thompson, W. B.	141, 706
Thorpe, C.	765
Timcenko, A.	909
Tomasi, C.	258
Tsai, R. Y.	850
Ulupinar, F.	544
Weems, C. C.	797
Weiss, R. S.	433, 650, 782
Williams, L. R.	639
Wixson, L. E.	631
Wohn, K.	405
Wolff, L. B.	162
Wolovich, W. A.	131, 919
Yang, D.	757
Yoshimi, B.	909

ACKNOWLEDGMENTS

The twentieth computer Image Understanding Workshop was held in Pittsburgh, Pennsylvania on September 11-13, 1990. This workshop, attended by more than two hundred research and government personnel, was sponsored by the Information Science and Technology Office of the Defense Advanced Research Projects Agency (DARPA).

Dr. Rand Waltzman, the DARPA Program Manager for Machine Vision, describes the main focus of the workshop as designed to increase awareness of Image Understanding research in the wider government, commercial and academic communities. The end result is to enhance the transfer of this technology to ongoing and projected programs in which computer vision may make a contribution. An important aspect, as is customary for these workshops, is to help keep the government research community aware of evolving technology, and most importantly, to exchange ideas, needs and trends in computer vision research. Professor Takeo Kanade, co-director of the Robotics Institute at Carnegie Mellon University, coordinated the technical program for this workshop.

The papers published in this proceedings were prepared by the research personnel of the various institutions involved in computer vision research. The first section is comprised of expanded research reviews presented by the principal investigators of the research institutions included in the DARPA program. The second section contains the technical papers prepared by members of the technical staffs of each institution arranged by major topics. Although these technical papers were not presented due to lack of time, they are included in order to present as full a record as is possible of recent results in the computer vision field. During the meeting, selected researchers presented in-depth reviews of important topics currently under research by member institutions.

The pictures appearing on the cover were submitted by Dr. Steven Shafer, a Senior Research Scientist of Carnegie Mellon University, in order to present a composite of several of the important research efforts being undertaken in image understanding at the University. Shafer describes each of the photos as follows:

Navigation: The NAVLAB autonomous truck drives itself on park roads and public side streets. Photo: Chuck Thorpe

Photointerpretation: An image database is used to register images to known landmarks, and stereo vision determines the 3D structure of terrain and buildings. Photo: Dave McKeown

Recognition: The Vision Algorithm Compiler uses the VANTAGE solid modeling system to automatically determine a vision program to recognize objects. Photo: Katsushi Ikeuchi

Vision Theory: Physics-based vision addresses geometry, optics, and uncertainty to measure object properties. Photo: Steve Shafer

Sensor Design: Perception research includes the design of new sensor technologies, such as this hybrid analog/digital chip for a high-speed light-stripe range sensor. Photo: Takeo Kanade

Manipulation: Vision algorithms and shape representation are dependent on the task to be performed, such as the use of a specific type of manipulator or gripper. Photo: Katsushi Ikeuchi

The artwork and layout designs were done by Ms. Rubina Ahmed of the SAIC graphics staff. Significant assistance in handling the mailings, and in putting the proceedings together for publication was provided by Ms. Inara Gravitis, Ms. Lynn Behnke, and Ms. Tu-Phuong Pham of the Science Applications International Corporation Staff.

SECTION I

PRINCIPAL INVESTIGATOR REPORTS

CMU Image Understanding Program

Takeo Kanade

Steven Shafer

School of Computer Science

Carnegie Mellon University

Pittsburgh, PA 15213

Abstract

The CMU Image Understanding Program covers a wide range of activities ranging from the study of basic computer vision science, to the development of visual sensors, to the demonstration of vision application systems. Highlights of the progress in this reporting period includes:

- Physics-based Vision
 - Color and Reflection
 - * Color Understanding
 - * Unified Reflection Model
 - * Interreflection
 - Shape, Motion, and Texture
 - * Robust Recovery of Shape and Motion from Image Sequence
 - * Stereo by Adaptive Window
 - * Texture Analysis by Image Spectrogram
 - High Quality Imaging
- Sensor Development
 - Fast Analog VLSI-based Range Finder
 - Photometric Sampler
- Vision for Object Recognition and Manipulation
 - Vision Algorithm Compiler
 - Rock Sampling
- Vision for Navigation
 - Navlab Progress
 - * Map Building
 - * SCARF
 - * YARF
 - * Architecture
 - * Integrated System Demonstration
 - Vision for Planetary Exploration
 - Terrain Modeling for Underwater Navigation
- Parallel Vision

1 PHYSICS-BASED VISION

Over the years, CMU vision researchers have been working on physics-based methods for computer vision that address modeling physical phenomena for robust and reliable low-level vision [Kanade, 1990]. The recent body of research

in this area has begun to conclusively demonstrate that algorithms derived from models of physical processes are far more accurate and reliable than heuristically derived algorithms. We have continued to make progress in this area for color understanding, unified reflection model, interreflection, texture analysis, and robust recovery of shape and motion from an image sequence

1.1 Color and Reflection

Color Understanding

For color understanding, CMU researchers have previously demonstrated separation of highlights from a color image [Klinker *et al.*, 1988] and color image segmentation by intrinsic object color [Klinker *et al.*, 1990]. Both of these are based on a dichromatic color reflection model [Shafer, 1985b] that Shafer developed, and their superior results have demonstrated clear advantage of the physics-based approach to color understanding which relies on *coherence of the physical model* rather than coherence of image color that traditional methods have relied upon.

One of our new developments is in the area of color constancy, which is predicting and matching object colors that are changed by variations in the color of illumination. We (Novak and Shafer) developed a new method we call "Supervised Color Constancy" in which a reference image of a known color chart is used to make an estimate of the spectral composition of the illumination to improve the accuracy of estimation of color changes [Novak and Shafer, 1990]. Previous methods for color constancy, which do not use a reference target, obtain a three-parameter description of the illuminant, and are subject to reliability problems if their heuristic assumptions are not met. With Supervised Color Constancy, we obtain typically 8 or more parameters to describe the illumination, and are free from heuristic assumptions.

In addition to the estimation of the illuminant, we developed a similar computation to actually predict color shifts of object colors; however, we haven't yet performed similar experiments with this algorithm. Finally, we propose a new paradigm we call "Incremental Color Constancy" suitable, for example, for a mobile robot travelling outdoors. In this paradigm, the color chart is not used; instead, the object colors estimated from previous frames are used as reference colors to estimate the illuminant in the next frame, which in turn is used to estimate the object colors of newly visible objects, and so on. We have not yet implemented this proposed technique. We are also continuing to study physics-based color reflection

tion analysis, and are now developing quantitative models of interreflection and surface properties [Novak *et al.*, 1990].

Unified Reflectance Model

Brightness of a pixel in an image results from the reflection of lights. Thus interpretation of images requires a sound understanding of the various mechanisms involved in the reflection process. Various reflectance models have been used in computer vision and graphics to deal with various types of surfaces. We (Nayar, Ikeuchi and Kanade) have been working toward a unified reflection model to describe reflection from surfaces that may vary from smooth to rough [Nayar *et al.*, 1990b] (in these proceedings).

There are two approaches to the study of reflection: physical and geometrical optics. While geometrical models may be construed as mere approximations to physical models, they possess simpler mathematical forms that often render them more usable than physical models. However, geometrical models are applicable only when the wavelength of incident light is small compared to the dimensions of surface imperfections. Therefore, it is incorrect to use these models to interpret or predict reflections from smooth surfaces. Only physical models are capable of describing the underlying reflection mechanism of such surfaces.

Nayar, Ikeuchi, and Kanade first consider the Beckmann-Spizzichino (physical optics) model and the Torrance-Sparrow (geometrical optics) model. They were chosen in particular as they have been reported to fit experimental data very well. The conditions that determine the validity of each model are carefully studied. From studying the behavior of both models, we propose a model comprising three reflection components: the *diffuse lobe*, the *specular lobe*, and the *specular spike*. The dependencies of the three components on the surface roughness and the angles of incidence and reflection are analyzed in detail. This general model is capable of describing reflection from surfaces that may vary from smooth to rough.

Shape from Interreflection

Since Horn [Horn, 1977], various techniques to extract shape from intensities based on photometric properties have been developed, such as shape-from-shading and photometric stereo. These methods, however, suffer when interreflections, also referred to as mutual reflections, occur. With concave surfaces, light rays will have multiple bounces, and points in the scene are illuminated multiple ways. In the presence of such interreflections existing shape-from-intensity methods produce erroneous results.

Interreflection has been studied extensively in computer graphics as a forward problem, that is, prediction of light brightness given the shape, reflectance and illumination. However, it has been almost untouched in computer vision as an inverse problem. We (Nayar, Ikeuchi and Kanade) have developed a technique called shape from interreflections that can recover the shape and reflectance of the scene in the presence of interreflections [Nayar *et al.*, 1990a].

The method presents a solution to the inverse problem for Lambertian surfaces of arbitrary (but continuous) shape, with possibly varying and unknown reflectance (albedo). The shape and reflectance recovery algorithm works as follows. First, a local shape-from-intensity method is applied to the concave surface to obtain "pseudo" (erroneous) estimates of

shape and reflectance. Our solution is based on the observation that the pseudo shape and reflectance, though erroneous, carry information about the actual shape and reflectance of the surface. The pseudo shape and reflectance are used to model the interreflection effects. We show that the pseudo shape is generally "shallower" than the actual shape and hence exhibits weaker interreflections. These interreflections are used to compensate the pseudo shape and reflectance estimates to obtain better estimates of shape and reflectance. This shape and reflectance is again used to model interreflections to obtain even more accurate estimates. Shape and reflectance estimates are iteratively refined to finally converge to the correct shape and reflectance. A detailed analysis of convergence is given for the simple case of two planar surface elements. Convergence for the more general case is discussed and demonstrated by numerous simulation results. Several experimental results of real objects demonstrate the robustness, accuracy, and practical feasibility of the proposed algorithm.

1.2 Shape, Motion, and Texture

Robust Extraction of Shape and Motion from Image Sequence

In principle, the shape of an object can be computed from a sequence of images by first estimating camera motion and depth, and then inferring shape from the depth values. In practice, however, when objects are distant from the camera, relative to their size, this computation is ill-conditioned. First, the translation component along the optical axis is difficult to determine, because the image changes that it produces are small. Second, shape values are very sensitive to noise if they are computed as the small differences between large depth values.

We (Tomasi and Kanade) have developed a theory [Tomasi and Kanade, 1990] (in these proceedings) that circumvents those difficulty by inferring shape directly from variations in the relative position of image features, without computing depth as an intermediate step. We show that shape and camera rotation can be inferred precisely from many features and frames, without assuming any model for the motion.

Our theory is based on the observation that the geometrical constraints due to incidence relations among projection rays can be expressed as the degeneracy of a matrix that gathers all the image measurements. To our knowledge, this observation has not previously appeared in the literature. We have shown that if noise were not included in the measurement, the matrix has rank 3. With noise, as is always the case in practical situations, the constraints can be precisely expressed by the concept of *approximate rank* [Forsythe *et al.*, 1977]. This theory reduces computation of shape and motion from an image sequence to decomposition of the measurement matrix.

The resulting algorithm, tested for the case of camera motion in 2D, gives very precise motion and shape estimates, without using any smoothing or relying on any motion constraints.

As an illustration of our theory, we used our algorithm to recover the shape of a one-dollar silver coin (about 4 cm in diameter) placed at 3.5 meters from a real moving camera with a long lens. The total rotation of the camera was 30 degrees around the coin (and in the midplane of the coin). The error in the computed angle of camera rotation was always less than a tenth of one degree, and usually substantially smaller. The

error in the shape of the coin was always less than 1.5 percent of its diameter, and typically considerably smaller.

Stereo by Adaptive Window

A central problem in stereo matching by computing correlation or sum of squared differences (SSD) lies in selecting an appropriate window size. If the window is too small and does not cover enough intensity variation, it gives a poor disparity estimate, because the signal (intensity variation) to noise ratio is low. If, on the other hand, the window is too large and covers a region in which the depth of scene points varies, then the disparity within the window is not constant. As a result, the position of maximum correlation or minimum SSD may not represent a correct estimate of disparity. For this reason, an appropriate window size must be selected locally. There has been, however, little research directed toward the adaptive selection of matching windows.

The stereo algorithm we (Kanade and Okutomi) propose selects a window adaptively by evaluating the local variation of the intensity and the disparity [Kanade and Okutomi, 1990] (in these proceedings). We employ a statistical model that represents uncertainty of disparity of points over the window: the uncertainty is assumed to increase with the distance of the point from the center point. This modeling enables us to assess how disparity variation within a window affects the estimation of disparity. As a result, we can compute the uncertainty of the disparity estimate which takes into account both intensity and disparity variances. So, the algorithm can search for a window that produces the estimate of disparity with the least uncertainty for each pixel of an image. The method controls not only the size but also the shape (rectangle) of the window. The algorithm has been tested on both synthetic and real images, and the quality of the disparity maps obtained demonstrates the effectiveness of the algorithm.

Texture Analysis

We have started a new effort in the area of texture analysis, and are now examining "space/frequency distributions" for vision. In particular, we (Krumm and Shafer) introduce the *image spectrogram* for machine vision [Krumm and Shafer, 1990] (also in these proceedings). The image spectrogram is a function that tells, for every point in the image, how much energy is present at each spatial frequency within the neighborhood of that point. The use of such space/frequency distributions is not completely new for machine vision – they have been used for 2D texture segmentation in the past. However, we are showing now how they can be used for 3D vision as well, to unify the analysis of various phenomena that have in the past always been studied independently. We begin with a re-formulation of shape-from-texture-gradients in the Fourier domain, which amounts to fitting a cubic function to the formants in the image spectrogram. Interestingly, this analysis does not require any traditional feature-finding such as edge detection before the 3D analysis. Next, we show how aliasing is caused by pixel resolution, and tie that in with the shape-from-texture theory to show the limits of 3D reconstruction. Aliasing causes artifacts in the image such as Moire patterns, which are very confusing to most vision methods. However, we show that with a computer-controlled zoom lens, the Moire patterns can actually be analyzed to yield a super-high-resolution image that shows spatial frequencies far above (i.e. finer than) the pixel resolution. This allows

much more accurate estimation of 3D surface shape. Finally, we examine the image properties that are related to precision (i.e. resolution) and accuracy (i.e. false matches) in stereo and motion analysis. We show that the image spectrogram, and the "repeatogram" derived from it, appear to capture much of the relevant information and might be used to evaluate the reliability of feature points for image-to-image matching. The general conclusion is that the space/frequency distribution makes it possible to unify very disparate phenomena in spatial vision, in particular showing how geometry-domain and Fourier-domain factors are interrelated and can be jointly analyzed.

1.3 High Quality Imaging for Computer Vision

Endemic to all quantitative vision analysis is the need for very high quality images. We have addressed this in two ways: by improving our hardware facilities, and by improving image quality using active camera control. Our hardware for physics-based vision is the Calibrated Imaging Laboratory (CIL), a unique facility for acquiring images in a controlled environment using an optical studio with specialized lighting, high-precision imaging equipment and motion platforms, and calibration instruments and targets. One of the distinguishing features of CMU vision activities is that we back up our theory by precise and quantitative experiments. The CIL has been the major instrument for this approach.

Since its founding in 1985 [Shafer, 1985a], the CIL has been one of the most comprehensive facilities for controlled experiments in machine vision. In the last year, we have moved to a much larger room, built high-precision automated lenses and motion platforms, and acquired a new high-precision CCD camera. We are currently building a "software control panel" to run all the equipment from a SUN workstation. As always, the CIL is available to interested researchers to come and collect data, by making arrangements with us (Shafer).

We (Willson and Shafer) also are investigating how to improve image quality by active camera control, using active control of lens parameters and camera motion to correct for undesirable optical artifacts [Novak *et al.*, 1990]. We have observed that for a wide variety of 35mm and video lenses, chromatic aberration causes noticeable mis-registration of images from one color band to the next. Using our high-precision, stepping-motor lens, we can now compensate for this by adjusting the zoom and focus each time we change color filters. In this way, we have reduced chromatic aberration in our automated lens from about a 3-pixel mis-registration to about 0.2 pixels, resulting in very noticeable improvements in such computations as color histograms. We are now studying "constant-magnification focusing", in which the magnification change induced by focusing is neutralized through active zoom control. Thus, as we focus the lens, we also zoom it minutely to keep the image at a constant magnification.

2 Sensor Development

A Very Fast Analog VLSI-based Range finder

Rangefinding, the measurement of the three dimensional profile of an object or scene, is a critical component for many robotic applications. We (Kanade, Gruss and Carley) have been developing a range finder (see figure 1), which can produce 100 to 1000 frames of range images per second by using

VLSI technology integrating photosensing and analog signal processing [Kanade *et al.*, 1989].

Among many different rangefinding techniques, light-stripe rangefinding is one of the most common and reliable methods. A conventional light-stripe rangefinder operates in a *step-and-repeat* manner — a stripe source is projected on an object, a video image is acquired, the position of the projected light stripe is extracted from the image, the stripe is stepped, and the process repeats. Though practical, range acquisition rates achievable using this method are severely limited; typically, on the order of one second is required to acquire a complete range image.

Geometrically, our range finder is based on the lightstripe range sensing method, but operationally it is different and time-based. Our rangefinder uses a specialized VLSI sensor which gathers range data as a scene is swept continuously by a moving stripe. The sensor consists of a two-dimensional array of smart photosensitive cells. Each cell has circuitry that detects and remembers the time at which it observed the peak incident light intensity during a sweep of the stripe. A given cell predefines a unique line of sight, and the recorded time determines a particular orientation of the plane. Thus, in a single pass of the light stripe over the scene, sufficient information is gathered to extract the ranges at all the pixels. Thus an entire range map is acquired in parallel, and the total time of acquisition is independent of the range map resolution, typically 1 to 10 msec.

The novelty of this approach is the use of integrated *smart* sensors, sensors which provide processing at the point of sensing by the use of VLSI technology — the ability to integrate photoreceptors, analog circuitry, and digital logic on a single CMOS die. After testing a few basic cell designs, we have fabricated several small arrays of smart cells (6×10 , 4×4 , and 5×5): see figure 2. The cell size is approximately $200 \mu\text{m} \times 200 \mu\text{m}$. The operation of those arrays has been confirmed in a rangefinder testbed with a laser and optics, and we could obtain signals from the chip that correspond to the sensing speed up to 800 frames per second. Currently we are fabricating a larger array (28×30).

Photometric Sampler

By making surface smoothness assumptions, the unified reflectance model described above is reduced to the hybrid model; a linear combination of Lambertian and specular components. Using the hybrid model, we (Nayar, Ikeuchi, Kanade) have developed a device called photometric sampler. The object surface is illuminated using multiple extended light sources, and a set of images, one for each illumination, is taken from a single direction. An extraction algorithm uses the set of image intensity values measured at each surface point to compute orientation as well as the relative strengths of the Lambertian and specular reflection components. At the last workshop we have reported a 2D version of the device [Nayar *et al.*, 1989]. The results have shown high accuracy in measured orientations and estimated reflectance parameters for Lambertian surfaces, specular surfaces, and hybrid surfaces whose reflectance model is composed of both Lambertian and specular components. In this period, we (Sato, Nayar, and Ikeuchi) have built a 3D version of the device (see figure 3), which is being tested for the use of surface inspection, such as IC wafers.

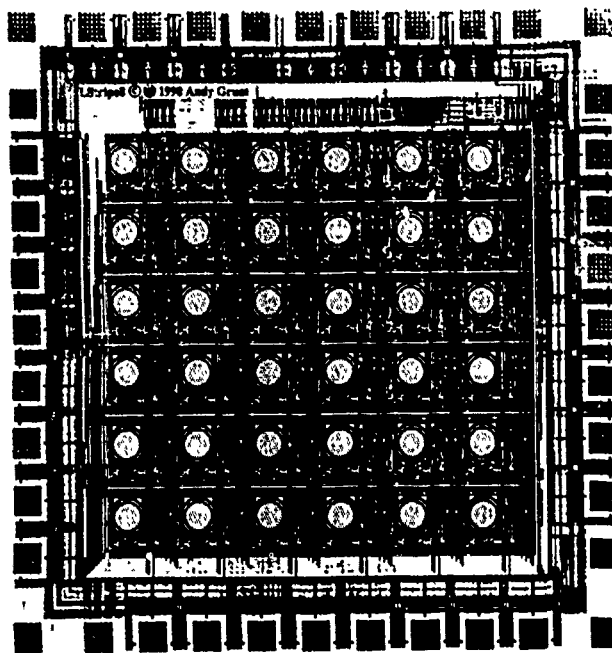


Figure 1: VLSI range sensor chip layout of a 5×5 array

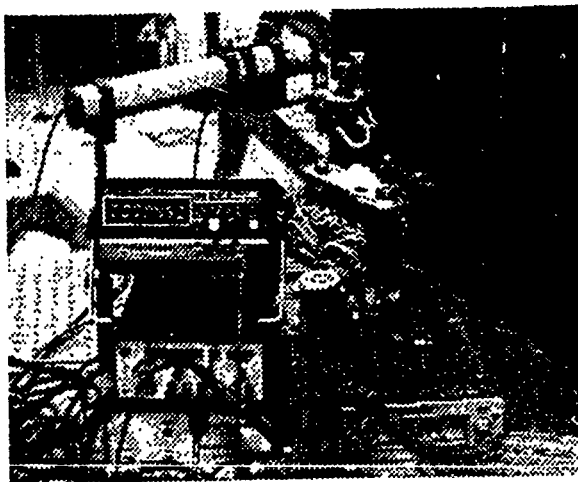


Figure 2: A range finder testbed. It consists of a laser scanner (on the right) and a camera optics (on the left). A VLSI sensor chip is mounted at the film plane of the camera.

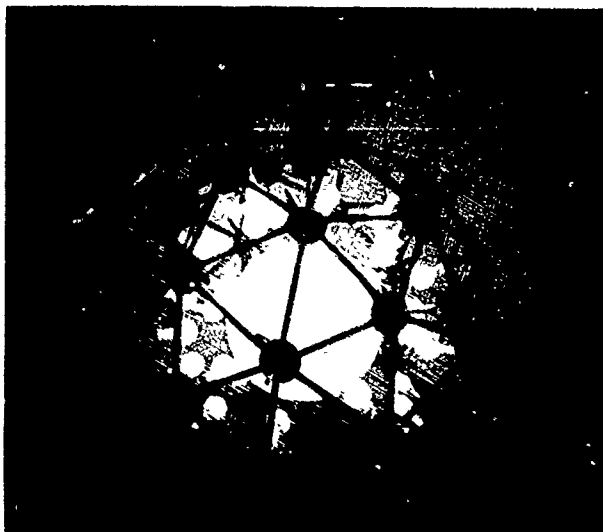


Figure 3: Photo of the 3D Photometric Sampler

3 Vision for Recognition and Manipulation

3.1 Vision Algorithm Compiler

A Vision Algorithm Compiler (VAC) is a technique that we have been developing over the years as an alternative to the traditional method of "hand-coding" model based vision systems [Ikeuchi and Kanade, 1988a]. Rather than requiring extensive time from highly skilled implementors, a VAC generates a strategy for a well-defined vision task automatically, given adequate models of objects, sensors, and processing techniques. Working off-line, a VAC analyzes models and determines a vision strategy. Later, an application system uses the strategy on-line to perform the specified task. We have worked on modeling sensors [Ikeuchi and Kanade, 1988b], a frame-based geometric modeling system [Kanade *et al.*, 1988], and an example system for bin-picking task [Ikeuchi and Kanade, 1988a].

We [Ikeuchi, Gremban, Wheeler, Hong] have been working on a compiler for the task of object localization (hence, we refer to our compiler as a VACL - Vision Algorithm Compiler for Localization). In an object localization task, the object is known, but its position and orientation are unknown. The VACL uses CAD models of objects, sensor models, and a predefined set of processing algorithms to generate a strategy.

We decompose the task of object localization into two distinct subtasks:

- aspect classification – An image of an object is classified into one of a small number of topologically distinct appearance groups called aspects. Each aspect represents a collection of viewpoints within which the object "looks roughly the same."
- linear shape change – Aspect classification yields only a rough estimate of the position and orientation of an object. However, this rough estimate can be used to ini-

tialize a more accurate procedure which matches model features to observed image features.

For aspect classification, we have previously developed a method to generate a classification strategy by performing recursive sub-divisions of possible aspects. However, such method does not generate an optimal strategy; it only generates a strategy among several possible strategies.

This year, we have developed a module that generates a minimum cost strategy for aspect classification [Hong *et al.*, 1990]. We can define a space of all possible classification strategies. Each entry in the space represents one strategy having a particular order of applying features for aspect classification. Given costs obtaining features and probabilities of occurrences of aspects, we can assign a computational cost to each entry (a strategy) in the space. The module searches through the space using a branch-and-bound method and selects the one that minimizes the expected computational cost of aspect classification.

For determining linear shape change, we have developed a module that precomputes the procedures needed to determine accurate position and orientation, given the initial estimate from aspect classification [Ikeuchi and Hong, 1989]. The module repeats the following process at each aspect. First, the module examines visible faces at each aspect and determines the most reliable face among them. Using the characteristics of the face, it determines a strategy to set up a local coordinate on the face; then, it embodies the procedures necessary to execute the strategy as well as the transformation between the local coordinate system and the body coordinate system of the object. Third, it embodies a procedure to match visible edges to visible model edges. Finally, it embodies a procedure to determine the position and orientation precisely, by solving edge-matching equation iteratively given edge correspondences by the previous procedure.

In the next year, for examining the VACL's applicability, we plan to apply VACL and its environment to several vision tasks such as SAR image recognition, generation of optimal sensor strategies and robot assembly task recognition system.

3.1.1 Sampling of Rocks

We (Hebert, Ikeuchi, Delingetter) have developed a vision and manipulation system to collect small rocks [Choi *et al.*, 1990] as an example of task-oriented vision [Ikeuchi and Hebert, 1990] (in these proceedings) for dealing with natural objects with less well-defined shapes. Our goal is to eventually integrate the system into the CMU Ambler, a six-legged autonomous robot for planetary exploration.

The rock sampling system we developed includes: a robot arm, a range finder, and a small terrain mock-up that contains sand and small rocks. The goal of the rock sampling system is to identify, locate, and pickup rocks from the terrain.

The perception subsystem uses a 240x256 range image of the scene as input. Three types of features are extracted from the range image: shadows, edges, and surface discontinuities. The features give an indication of where the boundaries of the objects may be located in the scene. The algorithm controls the growing of the boundary by modeling each feature as a generator of forces that attracts the boundary. Following the force field, the boundary moves towards the surrounding features. This approach allows us to locate objects in the scene even when only a very small number of visual features

are extracted from the image. This departs from other vision systems that implicitly assume that strong and reliable features can always be extracted, and therefore would not perform well in the type of unstructured environment that we are considering.

For representation of extracted objects, we approximate the set of 3-D points by a superquadric surface. The superquadric fitting algorithm is a gradient descent on the parameters of the surface.

We currently use a vertical manipulator that can translate in a X-Y plane, translate along the Z-axis (that is orthogonal to the scene), and rotate about the Z-axis. This configuration is suitable because the target vehicle, the Ambler, will provide the X-Y motion thus allowing for positioning of the effector precisely above the target sample. The current design of the gripper is a scaled down version of an excavation tool. This design allows for penetration of soft terrain, and it allows for other sampling operations such as scooping.

We have demonstrated the cycle of perception, representation, and manipulation on a variety of terrains. The experiments have shown that the system performs well even in the presence of difficult, unstructured terrain.

4 Vision for Navigation

4.1 Navlab Vision Progress

We (Thorpe, Hebert, and Kanade) have been working on the CMU Navlab, an integrated visual navigation system [Thorpe and Kanade, 1989] [Thorpe, 1990]. In this period we continue to make progress in advancing component capabilities, architecture, and system demonstration.

Map Building by Active Sensor

Use of active rangefinder (ERIM) images has been one of the major Navlab vision activities [Hebert and Kanade, 1988]. We (Hebert) have continued the development of a robust map building system using the ERIM range finder images for the Navlab [Hebert, 1989a]. The map building procedure is made robust by fusing information from multiple sources: 1) using the position information from the INS; matching well-defined discrete objects between frames before attempting to match terrain descriptions; and representing explicitly uncertainty and confidence to produce an accurate map and to remove spurious items from the map.

Matching objects is not very expensive in our case because we have only a few objects to match in each frame and because we can assume that we have a reasonable estimate of the displacement between frames from INS or dead-reckoning. One of the most difficult issues is to detect and remove spurious objects. Spurious objects appear in two cases: 1) noise in the range image causes the object detection program to hallucinate, and 2) moving objects (e.g. people) crossing the field of view are detected as objects. The problem of spurious objects is solved by calculating a confidence measure for each object, based on a sensor model and repeated observations of the object in its expected location.

The map building and matching is now integrated in the Navlab system and has been demonstrated in complex navigation scenarios.

SCARF

We (Crisman) have completed the work for SCARF, Supervised Classification Applied to Road Following, which tracks roads by adaptive color classification [Crisman, 1990]. SCARF runs in a loop of: classify image pixels, find the road model that best matches the classified data, and update the color models for classification. The models of road color and geometry used by SCARF make very few assumptions about the road, and make SCARF run robustly even when following unstructured roads.

SCARF represents multiple color classes, as Gaussian distributions in full RGB color, and calculates probabilities. SCARF typically uses four color classes to describe road appearance, and four for off-road objects. Pixels are compared to each of the eight classes, and are given both a label and a probability. In our experiments, any simplification of the system (using monochrome or color combinations, or using binary thresholds instead of probabilities, or assuming uniform variances for all classes) reduced the ability of the system to handle difficult situations, such as dirt roads, leaves on the road, and dark shadows.

SCARF uses a very simple model of road geometry. Roads are considered to be locally straight and flat. The only free parameters are the road's angle and horizontal offset relative to the vehicle. While this representation will not accurately represent curves or hills, it is relatively insensitive to minor misclassifications and local road imperfections. Our simple model allows rapid evaluation, and thus lets us build new road models as we drive, and compensate for curves or hills. The various versions of SCARF, some of which have been implemented on the Warp parallel supercomputer, have driven the Navlab along bicycle paths, dirt roads, gravel roads, and suburban streets, as well checking intersections.

YARF

Currently we (Kluge, Aubert, and Thorpe) are developing the YARF (Yet Another Road Follower) system. It explicitly models as many aspects of road following as possible, for driving on structured roads [Kluge and Thorpe, 1990] [Aubert and Thorpe, 1990]. Highways, freeways, rural roads, even suburban streets have strong constraints. Modeling these explicitly makes reasoning easier and more reliable. When a line tracker fails, for instance, an explicit model of road and shoulder colors adjacent to the line helps in deciding whether the line disappeared, became occluded, turned at an intersection, or entered a shadow. This kind of geometric and photometric reasoning is vital for building reliable and general road trackers.

YARF has individual knowledge sources that know how to model and track specific features, such as road edge markings (white stripes); road center lines (yellow stripes); and shoulders. YARF also uses an explicit geometry model of the road, consisting of location of vehicle on road; location of stripes; type of stripes (e.g. broken or solid); and maximum and current road curvature. Figure 4 shows tracking yellow and white lines through dappled shadows.

YARF is designed for higher speeds than SCARF, and runs in a more predictable environment. The combination of multiple trackers controlled by explicit models of road geometry, vehicle motion, and tracker performance, has allowed YARF to run the Navlab up to 15 kph using a single Sun 4. Future

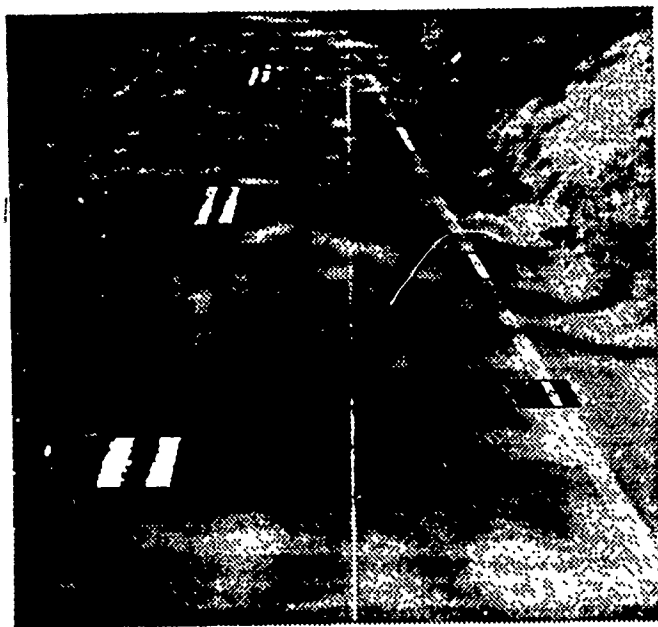


Figure 4: YARF tracking yellow and white lines in complex shadows

work will expand YARF to run on a multiprocessor, and will begin automatically interpreting tracker failures, to recognize and negotiate intersections and other scene phenomena.

EDDIE — Architectural Toolkit

Robots are physical problem-solving systems, and thus occupy a unique scientific and design niche and require unique processing and models. Robots are certainly physical systems, and robot designers can certainly borrow technology from physical systems such as signal processing and control theory. Many robots are also problem-solvers, and use ideas from symbolic systems, particularly Artificial Intelligence and high-level Image Understanding. But there remains an intermediate "robotics" level which is the peculiar domain of robotics research.

Our new EDDIE system (Efficient Decentralized Database and Interface Experiment) provides an architectural toolkit that supports this view of mobile robots. Rather than forcing an artificial standard for flow of control or data, EDDIE enables building specialized architectures for individual vehicles and applications. EDDIE begins with a new low-level controller [Amidi, 1990], which provides fast dead-reckoned position estimation and real-time trajectory tracking. Communications between modules, and to the controller, are fast, simple, point-to-point links with no central module bottleneck. Map issues are effectively divided into local and global representations.

EDDIE has been used to build several different architectures. The simplest systems use only a single perception module and the controller to do road following. The most complex systems we have built with EDDIE use several different road following modules, plus landmark detection, emergency stop,

and map position update, along with Annotated Maps for mission planning and execution.

In EDDIE, global, permanent, maps are handled by the separate mechanism of "annotated maps". Annotated maps start with a geometric representation of objects, such as roads, intersections, and landmarks. Annotations hold a wide variety of knowledge, both procedural and declarative, tied to a particular map location or object. Triggers are a special form of annotations, monitored by the EDDIE map manager. When the vehicle reaches the trigger's location, the map manager automatically sends a specified message to a named module.

Integrated System Demonstration

The most ambitious mission we have performed to date is a 0.4 mile run on unmodified suburban streets in Pittsburgh's North Hills. This run involved driving on curving streets; turning through intersections; stopping for unexpected obstacles; finding landmarks and updating vehicle position.

We built a map of the route, driving the Navlab by hand and using the laser scanner to record the location of 3-D objects. During the run, the vehicle started moving slowly, while it found landmarks to initialize its position. A trigger then caused the vehicle to speed up until it approached the first turn. At that point, triggers caused various modules to slow the Navlab, find 3-D objects, match them against the map, and update the vehicle's position estimate. Through the turn, vision was not able to see the road, so another trigger caused dead reckoning to take control until the vehicle was lined up with the next road, when the road was again in the field of view and vision could resume control. The run proceeded in this fashion until the final triggers, which matched the mailbox at the destination with the map, and brought the vehicle to a stop.

Others

Other progress includes: a new trajectory planner for cross-country navigation in rough terrain [Stentz, 1990]; rule-based road scene analysis [Fujimore and Kanade, 1990]; and recognition of cars in outdoor scenes [Kluge *et al.*, 1990]

4.2 Rugged Terrain Perception for Planetary Exploration

Under the sponsorship of NASA we have been developing a six-legged autonomous walker (Ambler) for planetary exploration [Bares *et al.*, 1989]. Research on its perception system has strong relationships to the Image Understanding Program, so it is briefly reviewed here.

There are two complementary aspects of the terrain that are important for locomotion: shape and material. Our research has concentrated on shape, with a smaller effort to identify material properties.

We describe terrain shape as an elevation map. We have implemented a mapping system that builds and maintains a composite elevation map, given (a) a sequence of laser rangefinder images and (b) the motion between images. In addition to the elevation, the system computes the elevation uncertainty, local slope, visibility, and foothold goodness (measure of terrain flatness in a foot-size neighborhood) [Caillas *et al.*, 1989].

Material properties of interest for locomotion include soil density, soil grain size, soil cohesion, and internal angle of friction. Using thermal imagery acquired by an infrared camera we have successfully demonstrated techniques to classify

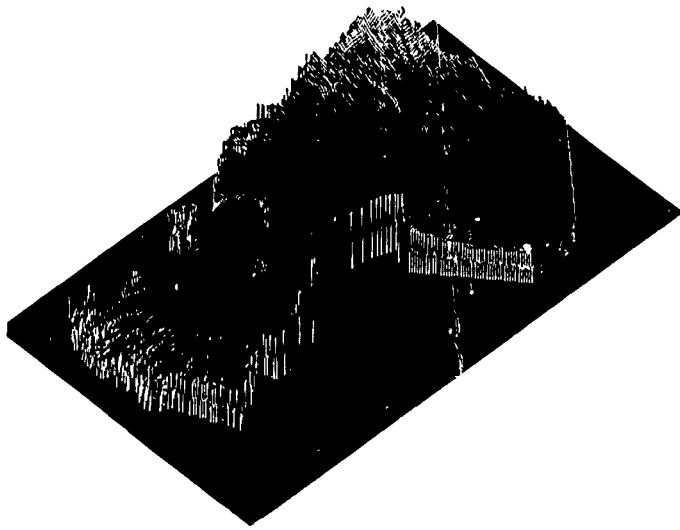


Figure 5: A composite terrain map

outdoor terrain regions as "sand" or "rock," based on a model that relates observed temperature to thermal inertia, and thermal inertia to soil density and grain size [Caillas, 1990]. Using force/torque sensing we have developed the capability to determine the stiffness and friction characteristics of terrain [Krotkov, 1990].

We have successfully demonstrated a pixel-based terrain matching algorithm to estimate the vehicle motion from a sequence of hundreds of range images. The matching procedure requires an initial estimate of the displacement, typically from an INS, and then uses that estimate to seed an iterative minimization procedure. Using the computed vehicle motion, we merge the range images into a composite map using maximum likelihood estimation. Figure 5 is the result of merging 125 images from the Martin Marietta ALV test site. We have also shown how to improve the accuracy of the composite map by incorporating prior information in the form of a coarse-resolution elevation map [Hebert *et al.*, 1989, Kweon and Kanade, 1990].

4.3 Terrain Modeling for Underwater Navigation

As the third application domain of autonomous navigation, we (Hebert and Langer) are also working on underwater terrain modeling for autonomous underwater vehicles (AUV) in collaboration with Florida Atlantic University. The research involves building maps from sonar images that can be used in the control of an AUV.

Our current work is divided into four parts [Hebert, 1989b]:

- Image processing: Sonar images require a significant amount of image processing in order to reduce the effects of medium attenuation, speckle noise, background noise, etc. We have developed techniques that enhance the image so that objects, shadows, and major terrain features can be detected.

- Object detection: We have developed a technique to extract objects from sonar images using the geometry of the acoustic shadows.
- Qualitative terrain modeling: In sonar images, it is more difficult to build a precise quantitative terrain representation due to the poor resolution and accuracy of the sensor. We are investigating a more qualitative representation of the terrain in which terrain regions are represented by bounds on their elevation and slope.
- Map building: Building maps from sonar images involves merging the qualitative maps generated from individual images.

The goal is to integrate map building system into an annotated map system and to demonstrate it on a vehicle currently being built by FAU.

5 Parallel Vision

We (Webb) are developing an architecture-independent method for programming both local and global image processing functions on parallel architectures. We have developed the *split and merge* programming model for these operations, and realized this model in the Adapt programming language.

In Adapt, the programmer breaks the algorithm down into four parts: *First*, a function that is executed before anything else; *Next*, a function that is executed once per pixel; *Combine*, a function that combines the results of two adjacent regions of rows of the image; and *Last*, a function that is executed after everything else. It can be shown that this programming model allows the implementation of any local or global operation that can be computed in forward or reverse order over the image.

We have implemented Adapt on a number of architectures: Warp (in two separate implementation, partitioning the image either by rows or by columns) and on Nectar, as well as on the Sun.

We have also implemented a number of different global image processing operations in Adapt, including histogram, connected components, minimum bounding rectangle, run-length encoding, quadtree generation, and so on. These operations allow us to compare the performance of Adapt with previous work; our results show remarkably good performance. For example, both histogram and connected components perform better in Adapt than in the original hand-coded Warp W2 programs, which were carefully optimized when they were originally written.

References

- [Amidi, 1990] O. Amidi. Integrated mobile robot control. Technical report, Robotics Institute, Carnegie Mellon University, 1990.
- [Aubert and Thorpe, 1990] D. Aubert and C. Thorpe. Color image processing for navigation: Two road trackers. Technical Report CMU-RI-TR-90-09, Carnegie Mellon, The Robotics Institute, 1990.
- [Bares *et al.*, 1989] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker. Ambler: An Autonomous Rover for Planetary Exploration. *IEEE Computer*, pages 18–26, June 1989.

- [Caillas *et al.*, 1989] C. Caillas, M. Hebert, E. Krotkov, I. S. Kweon, and T. Kanade. Methods for Identifying Footfall Positions for a Legged Robot. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pages 244–250, Tsukuba, Japan, September 1989.
- [Caillas, 1990] C. Caillas. Thermal Imaging for Autonomous Vehicle in Outdoor Scenes. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pages 651–658, Tsuchiura, Japan, July 1990.
- [Choi *et al.*, 1990] T. Choi, H. Delingette, M. Delouis, Y. Hsin, M. Hebert, and K. Ikeuchi. A Perception and Manipulation System for Collecting Rock Samples. In *Proc. NASA Workshop on Space Operations, Automation, and Robotics*, Albuquerque, New Mexico, June 1990.
- [Crisman, 1990] J. D. Crisman. *Color Vision for the Detection of Unstructured Roads and Intersections*. PhD thesis, Electrical and Computer Engineering Department, Carnegie Mellon University, May 1990.
- [Forsythe *et al.*, 1977] G. E. Forsythe, M. Malcolm, and C. B. Moler. *Computer Methods for Mathematical Computations*. Prentice Hall, Englewood Cliffs, NJ, 1977.
- [Fujimore and Kanade, 1990] T. Fujimore and T. Kanade. An approach to knowledge-based interpretation of outdoor natural color road scenes. In C. E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 4. Kluwer Academic Publishers, 1990.
- [Hebert and Kanade, 1988] M. Hebert and T. Kanade. 3-d vision for outdoor navigation by an autonomous vehicle. In *Proc. Image Understanding Workshop*, April 1988.
- [Hebert *et al.*, 1989] M. Hebert, T. Kanade, E. Krotkov, and I. S. Kweon. Terrain Mapping for a Roving Planetary Explorer. In *Proc. IEEE Robotics and Automation Conf.*, pages 997–1002, Scottsdale, Arizona, May 1989.
- [Hebert, 1989a] M. Hebert. Building and navigating maps of road scenes using an active sensor. In *IEEE Conference on Robotics and Automation*, pages 1136–1142. IEEE, May 1989.
- [Hebert, 1989b] M. Hebert. Terrain modeling for autonomous underwater navigation. In *6th International Symposium of Unmanned Untethered Submersible Technology*. University of New Hampshire, June 1989.
- [Hong *et al.*, 1990] K. Hong, K. Ikeuchi, and K. Gremban. Minimum cost aspect classification: a module of a vision algorithm compiler. In *10th Intern. Conf. on Pattern Recognition*, Atlantic City, N.J., June 1990. (a slightly longer version is available as CMU-CS-90-124).
- [Horn, 1977] B. Horn. Understanding image intensities. *Artificial Intelligence*, 8(2):201–231, 1977.
- [Ikeuchi and Hebert, 1990] K. Ikeuchi and M. Hebert. Task oriented vision. In *Proceedings of Image Understanding Workshop*, page in these proceedings. Morgan Kaufman, September 1990.
- [Ikeuchi and Hong, 1989] K. Ikeuchi and K. S. Hong. Determining linear shape change: Toward automatic generation of object recognition program. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, San Diego, June 1989. a longer version, containing programs, is available as CMU-CS-88-188.
- [Ikeuchi and Kanade, 1988a] K. Ikeuchi and T. Kanade. Automatic generation of object recognition program. *Proceedings of the IEEE*, 76(8):1016–1035, August 1988.
- [Ikeuchi and Kanade, 1988b] K. Ikeuchi and T. Kanade. Modeling sensor performance for model-based vision. In R. Bolles and B. Roth, editors, *Robotics Research: 4*, pages 255–263. MIT Press, Cambridge, 1988.
- [Kanade and Okutomi, 1990] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. Technical Report CMU-CS-90-120, Carnegie Mellon, School of Computer Science, April 1990. Also in these proceedings.
- [Kanade *et al.*, 1988] T. Kanade, P. Balakumar, J. Robert, R. Hoffman, and K. Ikeuchi. Vantage: A frame-based geometric modeling system. In *Proc. of Intl. Symposium and Exposition of Robots*, pages 1405–1420, Sydney, Australia, November 1988.
- [Kanade *et al.*, 1989] T. Kanade, A. Gruss, and L. R. Carley. A vlsi sensor-based rangefinding system. In *Proceedings of the 5th ISRR*, pages 383–390, Tokyo, Aug. 1989. ISRR.
- [Kanade, 1990] T. Kanade. Computer vision as a physical science. In R. Rashid, editor, *The 25th Anniversary of CMU Computer Science*, ACM Press Anthology, page (to be printed). Addison Wesley, 1990.
- [Klinker *et al.*, 1988] G. Klinker, S. Shafer, and T. Kanade. The measurement of highlights in color images. *International J. of Computer Vision*, 2(1):7–32, June 1988.
- [Klinker *et al.*, 1990] G. Klinker, S. Shafer, and T. Kanade. A physical approach to color image understanding. *International J. of Computer Vision*, 4(1):7–38, Jan 1990.
- [Kluge and Thorpe, 1990] K. Kluge and C. E. Thorpe. Explicit models for robot road following. In C. E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 3. Kluwer Academic Publishers, 1990.
- [Kluge *et al.*, 1990] K. Kluge, T. Kanade, and H. Kuga. Car recognition for the cmu navlab. In C. E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 6. Kluwer Academic Publishers, 1990.
- [Krotkov, 1990] E. Krotkov. Active Perception for Legged Locomotion: Every Step is an Experiment. In *Proc. IEEE Intl. Symposium on Intelligent Control*, Philadelphia, Pennsylvania, September 1990.
- [Krumm and Shafer, 1990] J. Krumm and S. Shafer. Local spatial frequency analysis for computer vision. Technical Report CMU-RI-TR-90-11, Carnegie Mellon, The Robotics Institute, 1990. also in these proceedings.
- [Kweon and Kanade, 1990] I. S. Kweon and T. Kanade. High Resolution Terrain Map from Multiple Sensor Data. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pages 127–134, Tsuchiura, Japan, July 1990.
- [Nayar *et al.*, 1989] S. Nayar, K. Ikeuchi, and T. Kanade. Extracting shape and reflectance of hybrid surfaces by photometric sampling. In *Proceedings of DARPA Image Understanding*, pages 563–583. Morgan Kaufmann, May 1989.

- [Nayar *et al.*, 1990a] S. Nayar, K. Ikeuchi, and T. Kanade. Shape from interreflections. Technical Report CMU-RI-TR-90-14, Carnegie Mellon University, Robotics Institute, 1990. Also, to be presented at Intern. Conf. on Computer Vision 90.
- [Nayar *et al.*, 1990b] S. Nayar, K. Ikeuchi, and T. Kanade. Surface reflection: Physical and geometrical perspectives. In *Proceedings of Image Understanding Workshop*, page in these proceedings. Morgan Kaufman, September 1990.
- [Novak and Shafer, 1990] C. L. Novak and S. Shafer. Supervised color constancy using a color chart. Technical Report CMU-CS-90-140, Carnegie Mellon University, School of Computer Science, June 1990.
- [Novak *et al.*, 1990] C. Novak, S. Shafer, and R. Willson. Obtaining accurate color images for machine vision research. In *Proc. SPIE Conference on Perceiving, Measuring, and Using Color*, Santa Clara, CA, Feb. 1990. SPIE.
- [Shafer, 1985a] S. Shafer. The Calibrated Imaging Lab under construction at CMU. In *Proc. DARPA Image Understanding Workshop*, page 509. SAIC, Dec 1985.
- [Shafer, 1985b] S. Shafer. Using color to separate reflection components. *Color Research and Application*, 10(4):210-218, Winter 1985.
- [Stentz, 1990] A. Stentz. Multi-resolution constraint modeling for mobile robot planning. In C. E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 11. Kluwer Academic Publishers, 1990.
- [Thorpe and Kanade, 1989] C. Thorpe and T. Kanade. Carnegie mellon navlab vision. In *Proceedings of DARPA Image Understanding*, pages 273-382. Morgan Kaufmann, May 1989.
- [Thorpe, 1990] C. E. Thorpe. *Vision and Navigation: the Carnegie Mellon Navlab*. Kluwer Academic Publishers, 1990.
- [Tomasi and Kanade, 1990] C. Tomasi and T. Kanade. Shape and motion without depth. Technical Report CMU-CS-90-128, Carnegie Mellon University, School of Computer Science, May 1990. Also, to be presented at Intern. Conf. on Computer Vision 90.

Image Understanding and Robotics Research at Columbia University

John R. Kender¹
Pater K. Allen
Terrance E. Boulton

Department of Computer Science
Columbia University, New York, NY 10027

0 Introduction

This report covers the research investigations of the Vision/Robotics Laboratory at Columbia University from April, 1989, to June, 1990. Three faculty members and 19 Ph.D. students are working on about a dozen projects in computer vision, with several of the projects involving other researchers in the department, in the university, or in the corporate research centers of AT&T, IBM, Philips, or Siemens.

The main body of this report details the goals, significance, and accomplishments of these projects, and follows this introduction. This introduction itself is an executive summary in two parts. The first part reviews some of the notable achievements of our graduate students over the past 14 months. The second part is a capsule outline of the rest of the report, summarizing our interests and results, together with the principal researchers associated with them. It includes not only "pure" image understanding work, but also our work on the vision-related aspects of assembly and navigation robotics.

0.0 Notable Graduate Student Achievements

At the 1989 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Larry Wolff published four papers [56, 57, 58, 59], placing Columbia second in the number of papers accepted.

At the 1990 IEEE International Conference on Robotics and Automation, Monnett Soldo's paper on visual robot navigation, "Reactive and Preplanned Control in a Mobile Robot" [42], was a finalist in the Best Student Paper competition.

In its nine years, the Vision/Robotics Laboratory has produced nine PhD students; the past year we are pleased to have graduated four. In alphabetical order, they are. Michalis Hatzitheodorou, "Shape from Shadows. Theoretical and Computational Aspects" [23], Mark Moerdler "Shape from Textures. A Paradigm for Fusing Middle-Level Visual Cues" [34]; Ajit Singh "Image-Flow Computation: Estimation-Theoretic Framework, Unification and Integration" [38]; and George Wolberg "Separable Image Warping: Implications and Techniques" [53].

0.1 Low-level Vision

0.1.1 Image Flow

A new framework classifies image-flow information into conservation and neighborhood information, and fuses them using estimation-theoretic techniques, allowing estimation of discontinuous flow-fields without blurring at motion discontinuities. It unifies and integrates the three prior approaches: gradient based, correlation-based, spatiotemporal energy-based, it produces the least mean squared error. (Ajit Singh [37, 38, 39, 40]).

0.1.2 Image Warping

A survey and analysis of different aspects of digital image warping, containing considerable amounts of C code, has been published. A novel data structure and algorithm does warping to and from arbitrary shapes. A new, highly efficient, general method (patent applied for) achieves 2-D image warps by separating into two successive 1-D warps; soon to be available in X-windows. Novel, sensor-based image restoration algorithms that are local and inexpensive have been designed and tested, exceeding existing ones. (George Wolberg, and Terry Boulton [51, 52, 53, 54, 55]).

0.1.3 Polarization and Physics-Based Vision

The POLARIS system uses polarization information to classify surface material, separate highlight/diffuse components of an scene, and compute local surface orientation. Polarization computations detect occluding contours on diffusely reflecting dielectrics. An approximation results in a near real-time system for material surface classification on the PIPE. The methodology applied to graphics yields more realistic imagery of reflecting surfaces. Additionally, photometric flow fields determine local surface orientation from a continuous variation of photometric stereo. Also, a new invariant in two-camera stereo allows the determination of the orientation of lines and surfaces, insensitive to baseline measurement error. Most recently, the photometric invariant of isophotes at parabolic points is shown to be the projection of a principal direction of zero curvature, enabling qualitative surface orientation analysis. (Larry Wolff, Dave Kurlander, and Terry Boulton [56, 57, 58, 59, 60, 61, 62, 63, 64]).

0.1.4 Color Contrast

Experiments in simultaneous color contrast contradict the predictions of traditional models, and specify constraints for neural network models. (Billibon Yoshimi, and Qasim Zaidi of the Psychology Department [65]).

0.1.5 Replication Lab

Several algorithms not invented here have been reimplemented for study. (Terry Boulton).

0.1.6 Languages for Sensors

"Sensor-C" is under development. (Terry Boulton).

0.2 Middle-Level Vision

0.2.1 Multiprocessor Surface Interpolation

Analysis and novel SIMD encoding of several existing algorithms for depth interpolation leads to two novel, optimal techniques that minimize interprocessor communication. (Dong Choi, and John Kender [17, 50]).

¹This work was supported in part by the Defense Advanced Research Projects Agency under contract N00039-84-C-0165

0.2.2 Shape from Texture Autocorrelation

Distortions in image autocorrelation provide a method for shape-from-texture, it runs in parallel on the Connection Machine. (Lisa Brown, and Haim Shvaytser of SRI Sarnoff Research Center [15, 16]).

0.2.3 Fusion of Shape from Textures

Based on the "augmented texel" data structure, a new method segments and classifies textures according to the relative contributions of independent texture methods to a fused texture percept. (Mark Moerdler [34]).

0.2.4 Shape from Shadows

An optimal algorithm for shape from continuous shadows runs in parallel, with provably minimal error. Regularization methods allow it to approximate as well as interpolate efficiently. Applied to laser light beams, the method puts a theoretic optimal foundation under light striping, better, it exploits shadow information. (Michalis Hatzitheodorou [23, 24, 25, 26]).

0.2.5 Dynamic Digital Distance Maps

The complexity of efficiently updating digital distance maps in dynamic, high-dimension, environments is analyzed, and the method is implemented. (Terry Boulton [10]).

0.2.6 Generalized Cylinders

The contour image of a straight homogeneous generalized cylinder is shown to leave two, and only two, parameters unconstrained. A method for ruling the contour image allows the recovery of object tilt and axis location, given the additional information of intensity values along extremal cross-section curves. (Ari Gross, and Terry Boulton [18, 19, 20, 21, 22]).

0.2.7 Energy-Based Surface Segmentation

A sequential energy-based segmentation system allows testing of different grouping heuristics, implementation on a Connection Machine is nearly complete. The energy computation is novel, based on reproducing kernel splines, and using more robust (L-1 and L-infinity) error measures, it is especially efficient for sparse data. (Terry Boulton, and Mark Lerner [11, 12, 13, 14]).

0.2.8 Flexible Extruded Objects

A non-iterative transform applied to range data derives the radius of flexible extruded objects, such as wires and tubes. The other six degrees of freedom (position, orientation, curvature) are derived by the system from novel Hough-like parameter spaces: a total of seven dimensions, but efficient and self-limiting to noise. (John Kender, and Rick Kjeldsen of IBM T.J. Watson Research Center [29]).

0.3 High-Level Vision and Systems

0.3.1 Robust Visual Robot Navigation

Algorithms for the representation of space and free-space path planning have been surveyed. Preplanned and reactive control are integrated in an architecture that achieves robust autonomous navigation in an AT&T mobile robot, in over 100 runs. Its edge-tracking filter, and its mechanism for recording ground truth are also novel. (Monnett Soldo [41, 42, 43, 44]).

0.3.2 Model-Based Active Sensing

Haptic recognition via active exploration with a instrumented robot hand is achieved by combining geometric constraints, interpretation tree methods, and exploratory moves. Applicable to vision, constraints from paired line segments drive cost functions to determine the next sensor move. Optimal rotational parameters are developed using quaternions. (Ken Roberts [35, 36]).

0.3.3 Vision Planning

The complete locus of camera poses and optical settings that satisfy visibility, field-of-view, resolution, and focus requirements for a given object have been analytically determined and implemented; coverage and efficiency greatly exceed existing generate and test methods. (Dino Tarabanis, Peter Allen, and Roger Tsai of IBM T.J. Watson Research Center [45, 46, 47, 48, 49]).

0.3.4 Topological Navigation by Landmarks

Topological visual navigation in two-dimensional spaces by following directions to landmarks has been formalized and implemented with a hand-held camera. Navigators seek on topology, and adjust on symmetry. (John Kender, Avraham Lefi, Il-Pyung Park, and David Yang [27, 30, 31, 32]).

0.3.5 Visual Servoing

Objects moving with arbitrary trajectories are tracked by the PUMA arm in real-time, using PIPE visual input, a video is available Real-time grasping of moving objects by the Utah hand is under development. (Peter Allen, Aleksandar Timcenko, and Billibon Yoshimi [5]).

0.3.6 Integrated Hand-Eye Systems

A hand-arm system controlled by a task description language grasps and manipulates objects based on a robust line-based stereo system that recovers 3-D axes of surfaces of revolution. (Peter Allen, Paul Michelman, and Ken Roberts [1, 2, 3, 4, 6, 7, 8, 33]).

0.3.7 Sensor-based Volume Registration and Recovery

In an initial implementation, the PROVER system numerically recovers the parametric representations of volumes from many types of sensor data. The causes of misregistration in medical volume data has been surveyed. (Lisa Brown, and Terry Boulton).

0.3.8 Bayesian Analyses for Image Interpretation Systems

A novel bayesian network strategy is under development (Michelle Baker, and Terry Boulton [9]).

We now detail these efforts, many of which are documented by full papers in these proceedings. In each of three major groupings, the research is presented in order of relative maturity: from dissertations and full papers, through work in progress, to nascent investigations.

1 Low-level Vision

1.1 Image Flow

Visual motion is a major source of three-dimensional information. It is commonly recovered from time-varying imagery in the form of a two-dimensional image flow field.

A new framework classifies the image flow information available in time-varying imagery into two categories-- conservation information and neighborhood information. Each type of information is recovered in the form of an estimate accompanied by a covariance matrix. Image flow is then computed by fusing the two estimates using estimation-theoretic techniques. This framework is shown to allow estimation of certain types of discontinuous flow fields without any a-priori knowledge about the location of discontinuities; flow fields estimated using this framework are not blurred at motion-discontinuities. Two algorithms based on this framework have been analyzed and implemented, and the results of applying these algorithms to a variety of image sequences have been evaluated and discussed. In order to put the framework in context of an application, the image flow fields recovered by these algorithms are used in a Kalman filtering approach to incrementally estimate the scene depth.

The new framework is shown to be applicable identically to each one of the three major approaches for recovering conservation information, i.e., gradient-based approach, correlation-based approach and spatiotemporal energy-based approach. The formulation of neighborhood information used in this framework is also shown to reduce to some of the existing smoothing-based formulations under various simplifying assumptions. Thus, the framework described unifies various existing approaches for image flow computation. Such unification is useful in analyzing various existing frameworks as well as in generating new frameworks.

This new framework is also shown to serve as a platform to integrate the three approaches mentioned above. The measurements obtained by the three approaches have different error characteristics. This situation is analogous to the multi-sensor fusion problem, where the algorithms based on the three approaches behave as multiple sensors measuring image flow. An integrated framework applies the principles of statistical estimation theory to fuse the measurements obtained from different approaches. The resulting estimate of image-flow has the minimum mean-squared error. Other novel algorithms based on this framework have also been described.

(Ajit Singh [37, 38, 39, 40]).

1.2 Image Warping

The existing technology for image warping is extensive, but relatively slow, constrained by numerous side conditions and subject to many errors and aberrations. A search for better algorithms resulted first in a comprehensive survey of digital image warping techniques, containing considerable amounts of C code; it has been published as a monograph.

The literature is largely silent on the problem of efficiently and smoothly mapping between two image regions which are delimited by arbitrary closed curves; such regions do not have the universally assumed four corners. A second result was the specification and verification of an algorithm that instead treats an image region as a collection of interior layers around a skeleton. These layers impose a type of local polar coordinate system which allows each shape to be "unwrapped" into a tree-like representation. Region-to-region warping is then defined by a natural mapping between the two resulting trees. Although there is no a priori way of defining quality of mapping, the results are esthetically pleasing.

A third product is a new, highly efficient, general method for achieving 2-D image warps by separating the 2-D transform into two successive 1-D warps. It therefore extends the power of existing hardware systems that perform more limited classes of transformations by similar decompositions. However, this method shows that off-the-shelf hardware, in the form of digital filters with only minor modification for 1-D image resampling, can be used to realize arbitrary mapping functions cheaply and at video rates. The first release of the software embodiment of such an image warping system has been completed, and is now being ported to the X-Window environment. In addition, an interactive interface has been developed to allow for easier user definition of the warping functions. A patent on the system is pending.

Lastly, in analyzing the errors of the system, it was discovered that a major problem was in image reconstruction: the warping accesses the original image at spatial resolutions that differ significantly from the original sampling rate. In seeking better reconstruction filters, two new constraints on image reconstruction were discovered: the identity imaging constraint, and the idea of imaging-consistent reconstruction.

The identity imaging constraint requires that the magnification/minification filters used in a warping system should form a transform pair, such that magnification followed by minification back to the original size should not introduce errors. Surprisingly, this constraint is not satisfied by most image reconstruction filters, with the exception of sinc and box filters. The novel restoration/reconstruction filters invented in this effort do satisfy this criteria, however.

The idea of imaging-consistent reconstruction is a bit more complex. It stems from the need to incorporate a model of the imaging sensor into the reconstruction process; this requires functional forms for the reconstruction, rather than discrete convolutions commonly used. To develop such low error, local, functional filters, it is necessary to model the blurring function of the sensor, for example, the cell blurring in a CCD. Given this model, an analysis derived from the results of information-based complexity creates a local and efficient approximate image restoration resulting in a functional form blurred according to the camera model. The analysis first chooses one of the infinitely many scenes which could have generated the original image, and then defines the reconstruction as the blurry form of that one possible scene. Thus, restoration results are exact for a scene which could not be distinguished from the original scene given the digital image. Reminiscent of backwards error analysis, this is the idea of imaging-consistent reconstruction.

It is not hard to show that the error of these reconstruction algorithms are within a factor of 2 of the optimal algorithm. Further, assuming that the camera model is the filter for image minification, then the local restoration algorithms exactly satisfy the image identity constraint. In general, however, the reconstructions have been blurred, hence they do not satisfy the constraint exactly, but for many camera models they satisfy it approximately, except near very strong edges.

Assessing the quality of our new filters using traditional measures of the quality of image reconstruction filters, such as spectral analysis, these new filters prove superior to previous local methods of similar support, and rival or surpass the global method of cubic-spline interpolation.

(George Wolberg, and Terry Boult [51, 52, 53, 54, 55]).

1.3 Polarization and Physics-Based Vision

Research on the polarization of light by surfaces has resulted in the demonstration of POLARIS, an integrated system that uses polarization information to classify surface material, separate highlight/diffuse components of an scene, and compute local surface orientation. Derived from the Torrance-Sparrow theory of reflection and the Wolf polarization theory of "quasi-monochromatic" light, the new methods classify material surfaces into conductors or dielectrics by computing an empirical determination of the polarization Fresnel ratio. Specular regions have been identified on both metals and dielectrics on a per-pixel basis without the use of a segmentation procedure, the method only requires a controllable polarizing filter placed between the camera and the object. Further relationships implicit in the equations have been exploited in order to determine local surface orientation properties. Lighting need not be restricted to point sources, the methods have been extended to more typical extended sources, such as fluorescent tubes. Experimental testing of these initial components have given strong results, supporting the claims for the approach's utility.

The system has been extended both in theory and in practice. In theory, the Fresnel reflectance model now incorporates the polarization properties of diffuse reflecting bodies, which were originally assumed to be non-polarizing. The result is the prediction of the ability to use percent polarization computations to detect occluding contours on diffusely reflecting dielectrics. Together with prior observations that many albedo edges do not cause significant polarization edges, this indicates that polarization information is a powerful filter for finding edges due to occlusion, while eliminating edges due to albedo.

In practice, a near real-time approximation for material surface classification has been demonstrated on the PIPE image processing engine, requiring 6-30 frames (depending on the desired accuracy). The algorithm is simple enough that it should run on almost any frame rate pixel processor. A related approximate algorithm can do near real-time detection of occluding contours on diffusely reflecting dielectrics.

This understanding of polarization has been also been applied to graphics, resulting in more realistic imagery of mutually reflecting surfaces, such as building windows seen reflected in a lake, or convoluted objects like vases.

Separate in scope, but also derived from a detailed examination of the physics underlying the image-forming process, are two results in stereo. Local surface orientation and curvature can be derived from the "photometric flow field", which is the rate of change in the image irradiance of the image with a stationary object, a stationary camera, but a moving light source. The method is a generalization of (discrete) photometric stereo; instead of three light positions, the illumination geometry is allowed to move smoothly in three-space. Surface orientation has also been demonstrated from the stereo correspondence of linear features; the method computes the orientation of a plane from the orientations of two or more coplanar lines. Compared to point-based stereo, errors with respect to camera baseline translation or with respect to object distance from baseline grow much more slowly, as shown by both analysis and Monte Carlo simulation.

Most recently, differential geometry, in the form of the differential of the Gauss map (the "surface centered curvature matrix"), has yielded a new shape-from-shading invariant. The isophotes, that is, the image curves of equal reflected radiance, corresponding to the parabolic points of a surface are shown to be the projection of a principal direction of zero curvature. Under general conditions, the result is invariant to both reflectance function and viewer position, thus providing a powerful cue for qualitative surface orientation analysis.

(Larry Wolff, Dave Kurlander, and Terry Boult [56, 57, 58, 59, 60, 61, 62, 63, 64]).

1.4 Color Contrast

In conjunction with the Department of Psychology, color contrast phenomena have been investigated, with challenging results.

If a colored patch is surrounded by a larger patch of another color, the appearance of the enclosed patch can change markedly. Two types of mechanisms are thought to mediate this phenomenon: mechanisms that enhance contrast across contours, and mechanisms that integrate inside closed contours to give a uniform appearance. These mechanisms were studied by independently varying the area, perimeter length, and shape of the enclosed test patches.

For light-dark modulations of the surround, tests with equal area showed essentially the same amount of color induction. There was a slight increase in the amount of induction as the perimeter length was more than doubled, and there was also a tendency for the shapes with fewer lobes to show more induction than shapes with more lobes. For isoluminant red-green and yellow-blue surround modulation, the slopes of the functions relating induction to perimeter length were slightly steeper and the shapes with fewer lobes showed a larger difference from the shapes with fewer lobes. These results contradict the predictions of traditional induction models like the edge-distance model, and they specify constraints for the integration stage of network models.

(Billbon Yoshimi, and Qasim Zaidi of the Psychology Department [65]).

1.5 Replication Lab

A replication lab, just begun, has reimplemented and is in the process of evaluating algorithms for: segmentation techniques with Markov random fields, Kalman-filter based depth from translational motion, and multi-sensor registration techniques.

(Terry Boult)

1.6 Languages for Sensors

A language development effort for "Sensor-C" has started. This language extends C++ to allow for efficient and easy use of sensor type objects, currently not well supported. This work is in conjunction with the PROFIT language project in the Department of Computer Science.

(Terry Boult and Gail Kaiser).

2 Middle-Level Vision

2.1 Multiprocessor Surface Interpolation

Many constraint propagation problems in early vision, including depth interpolation, can be cast as solving a large system of linear equations where the resulting matrix is symmetric, positive definite, and sparse. Analysis and simulation of several numerical analytic solutions to these equations for a fine grained SIMD machine with local and global communication networks (e.g., the Connection Machine) shows that two methods are provably optimal in terms of computational complexity. For a variety of synthetic and real range data, the adaptive Chebyshev acceleration method executes faster than the conjugate gradient method, if near-optimal values for the minimum and maximum eigenvalues of the iteration matrix are available.

When these iterative methods are implemented in a pyramidal multigrid (coarse-medium-fine) fashion, using a fixed multilevel coordination strategy, the multigrid adaptive Chebyshev acceleration method executed faster than the multigrid conjugate gradient method again. This appears to be the case because an optimal Chebyshev acceleration method requires local computations only. These methods have now been validated on actual range data.

(Dong Choi, and John Kender [17, 50]).

2.2 Shape from Texture Autocorrelation

A new method for determining local surface orientation has been developed from rotationally invariant textures based on the two-dimensional two-point autocorrelation of an image. This method is computationally simple and easily parallelizable, uses information from all parts of the image, assumes only texture isotropy, and requires neither texels nor edges in the texture. Applied to locally planar patches of real textures such as roads, dirt, and grass, the results are highly accurate, even in cases where human perception is so difficult that people must be assisted by the presence of an artificially embedded circular object. The method runs on the Connection Machine.

(Lisa Brown, and Haim Shvaytser of SRI Sarnoff Research Center [15, 16]).

2.3 Fusion of Shape from Textures

Existing work on the fusion of five different shape-from-texture methods has suggested a novel approach for classifying textures. Each of the methods is tuned to certain image phenomena; the five are shape from spacing, shape from orientation, shape from size, and shape from absolute and relative eccentricity. Given a single texture patch, particularly one under perspective, each method will respond differentially according to the degree it believes the patch possesses cues that the method can exploit to derive shape information. These differential strengths can be gathered together in a new data structure, the "augmented texel", as a signature feature vector for the texture, where they can be manipulated in the usual way by standard pattern recognition or image segmentation techniques.

(Mark Moerdler [34])

2.4 Shape from Shadows

The determination of surface shape from the location of shadow boundaries has been extensively analyzed, particularly in the setting of continuous mathematics. This has led to an optimal, parallelizable algorithm, demonstrated on a network of workstations. The problem is decomposable into a series of one-dimensional slices in the plane of the moving light source; each slice admits of a fine-grained parallelism, whereas the decomposition itself can be attacked via coarse-grained parallelism. Each strip is computed using as a basis a family of interpolating splines of an unusual piecewise linear form. The solution is checked against a side system

of inequalities; if the solution fails, a non-linear approximation algorithm accommodates the failing constraints.

More recently, a smoothing spline approach has been developed to regularize noisy data; hence the algorithm can approximate data rather than interpolate it, with a corresponding increase in robustness. Empirical investigations on natural objects have demonstrated that the method is relatively insensitive to a range of smoothing parameters.

Additionally, the question of how to optimally position the illuminants was solved in some very restricted cases: the tangent of the incoming light ray angle should be uniformly distributed from zero to the maximum tangent permitted. As an offshoot, this investigation yielded a theoretic foundation for optimal laser light striping algorithms. The theory suggests and demonstrates that the shadows in light striping, ignored up to now, can instead be exploited to further increase accuracy.

(Michalis Hatzitheodorou [23, 24, 25, 26]).

2.5 Dynamic Digital Distance Maps

Spatially varying distance cost problems, such as path planning under the considerations of surface height or terrain quality, are relatively frequent, but vertex-based algorithms do not generalize well to these problems. Complexity bounds have been derived on the constrained distance transform for computing digital distance maps; they have also been extended to handle path planning with spatially varying distance metrics.

(Terry Boulton [10]).

2.6 Generalized Cylinders

Straight homogeneous generalized cylinders (SHGCs) are a flexible class of parametric shapes capable of modeling many real-world objects. An image invariant has been demonstrated that quickly and cheaply tests an image for the probable presence of a SHGC, under various rotational transformations and imaging conditions.

Building on this prior work, under a slightly more general definition of SHGCs, it has been determined exactly what constraints can be derived directly from their image contours. Given most conditions, all the parameters of an SHGC are shown to be recoverable from contour, except two, and only except two, the tilt toward the viewer, and the translation of the axis in the viewer direction. Therefore, every general orthographic view of an SHGC can be shown to have resulted from any member of a two-parameter infinite family of SHGCs which are contour equivalent. Within this family, the shape can vary significantly, even to the point of reversing the sign of the gaussian curvature. Implementing these results, a method for ruling SHGC contours has been demonstrated, once the image has been ruled, all those parameters derivable from contour alone can be recovered.

Faced with a lack of two constraints, either the use of more image information or the use of shape heuristics is necessary. Under the assumption of constant, pure diffuse reflectance, it has been determined that a non-monotonic SHGC can be recovered from a single intensity image. The tilt of the object is recovered by using the ruled contour image and intensity values along extremal cross-section curves. The location of the object's 3D axis is recovered from intensity values along meridians of the surface. This recovery algorithm has been tested on synthetic images, and is being reimplemented for real imagery. The incorporation of heuristics based on human perception of shapes, particularly the apparent human preference for representational economies such as object symmetry and object orthogonality, is under development.

(Ari Gross, and Terry Boulton [18, 19, 20, 21, 22]).

2.7 Energy-Based Surface Segmentation

Prior work has demonstrated that energy-based surface segmentation, derived from the optimal mathematics of information-based complexity, is inexpensive, local, easily parallelized, rapidly updatable in the presence of change or noise, and more accurate

than other existing (and slower) methods. Recently, three major advances have been achieved in this work on regularized surface reconstruction.

First, an initial version of the sequential energy-based segmentation system has been implemented, it serves as a testbed to analyze how different grouping heuristics reduce the computational complexity of surface reconstruction and segmentation. A parallel implementation on a Connection Machine is nearing completion.

Secondly, how to compute the energy under a wide, two-parameter range of class/norm assumptions has been demonstrated. The energy computation is based on reproducing kernel splines, and again is efficient, especially for sparse data, and is amenable to parallel implementation.

Third, the efficient computation of two types of "robust" smoothing splines has been derived. They are unlike the traditional smoothing splines used in regularization. Based on the relationship between reproducing kernel splines and covariance measures, regularization splines (L-2 norm splines) are shown to be a type of least square estimate: they are optimal bayesian estimators for the data, under the assumption that the errors are normally distributed. However, smoothing splines based on L-1 or L-infinity norms are known to be more robust with respect to other, more realistic, assumptions on the errors. Although a general theory for these robust smoothing splines has long existed, the techniques for generating them require general non-linear minimization, which can be very expensive. The newly discovered technique, however, reduces the problem in these useful cases to the solving a linear programming problem, which is far more efficient.

(Terry Boulton, and Mark Lerner [11, 12, 13, 14]).

2.8 Flexible Extruded Objects

A Hough-like parameter space technique for modeling flexible extruded objects as piecewise toroidal has been analyzed, and a novel transform has been implemented that derives their three-space curved axes from position and surface normal information. The method is purely local, and succeeds where attempts to model objects as being piecewise cylindrical fail. Although the local computation involves 15 free variables (for three points each, three of position, two of orientation), does not involve the iterative solution of non-linear equations. It has been demonstrated on synthetic and real range data.

Because the torus is an object with seven free parameters, this work also has demonstrated the robustness of the parameter space approach, even for high order objects. Better, it has demonstrated that the structure of the parameter spaces themselves can be chosen to counteract the triangulation error. Errors that occur in trying to find tori in objects that are unusually large or small, or unusually straight or flexed, can be made self-limiting.

This work required the extensive use of a symbolic mathematical analysis system (IBM's proprietary Scratchpad II), the resulting transform is based on a quadratic equation whose coefficients incorporate 12 inner products of three-space vectors. Along the way, it was discovered that, under some fairly general conditions, every torus has a large family of anti-tori. Their hallucinatory appearance in the image must be explicitly ignored.

(John Kender, and Rick Kjeldsen of IBM T.J. Watson Research Center [29]).

3 High-Level Vision and Systems

3.1 Robust Visual Robot Navigation

Algorithms for the representation of space and free-space path planning have been surveyed. The survey also proposes a taxonomy of this new field. There continues to be a relative paucity of results on qualitative, topological navigation, however.

Partly to address that need, an architecture that integrates preplanned and reactive control to achieve smooth, natural, autonomous navigation in a mobile robot has been demonstrated, in

over 100 runs. The robot, now donated to Columbia by AT&T, is an autonomous three-wheeled indoor vehicle equipped with odometry, ultrasonic sensors, and a camera augmented by a proprietary real-time vertical edge finding chip. Control of the robot is distributed among a set of behavior experts that tightly couple sensing and action. Input is neither fused nor compared against a prestored map. Instead, collections of behavior experts define global behaviors for the robot, and those behaviors are composed into plans that direct navigation.

The work has demonstrated novel techniques for robust real-time sensing, including an interesting variation on Kalman filtering. It also records a significant advance in the methodology of mobile robot experimentation and evaluation. To record ground truth as a function of time and video input, the robot pulls a small trailer holding a split-screen dual camcorder. The camcorder thus records on one video tape the time stamp, the robot video input, and the video image of the floor directly below the robot, on which fiducial markings have been previously applied.

(Monnett Soldo [41, 42, 43, 44]).

3.2 Model-Based Active Sensing

In work closely related to visual sensor planning, an integrated system of active touch strategies identifies polyhedral 3-D objects through exploration. This work combines three approaches. It uses geometric constraints between components to eliminate interpretations, it invokes interpretation tree methods for choosing the best active sensing move, and it plans and moderates exploratory moves made by tracing the object surface. A new constraint involving pairs of line segments has been developed, it is directly applicable to visual sensing. The choice of active sensing move is determined by a generic cost function, also applicable to the planning of mobile visual platforms.

The placement of multiple fingertips to specified 3-D world locations is an underconstrained problem. Methods have been developed for generating good generic candidate grasps to be optimized. A successor algorithm has been demonstrated which, given a desired position and normal for each fingertip, then computes all the joint angles for the fingers and arm by optimizing an intriguing cost function. Optimal rotational parameters are chosen by several new techniques using a quaternion representation. These later methods could also be used for representing and analyzing the visual pose of an object.

(Ken Roberts [35, 36]).

3.3 Vision Planning

Techniques have been developed, and a system, MVP (Machine Vision Planning), has been built to analytically determine the complete locus of camera poses and optical settings for viewing a given feature of an object. First, for each viewing constraint in isolation, admissible domains of sensor locations and settings are determined, by the analysis of the effect of the constraint on the CAD-CAM representations of the object. Then these component results are combined in order to find globally admissible sensor parameter values. Current techniques analytically satisfy visibility, field-of-view, resolution, and focus requirements; others are under development.

The approach is more accurate and far more efficient than existing generate-and-test sampling techniques. Camera placement experiments are available on a video tape that demonstrates the method in an actual robotic setup.

(Dino Tarabanis, Peter Allen, and Roger Tsai of IBM T.J. Watson Research Center [45, 46, 47, 48, 49]).

3.4 Topological Navigation by Landmarks

A model for topological visual navigation in two-dimensional spaces has been formalized and implemented. It explores and emphasizes the methods and the efficiencies of qualitative visual descriptions of objects, and of direction-giving by means of visual landmarks.

The model formalizes three domains--the world itself, the map-maker's view of it, and the navigator's experience of it--and the concepts of custom maps and landmarks. Visual landmarks are shown to be chosen depending on which of several costs (sensor, distance, communication, or others) should be minimized; paths minimizing one measure can make others arbitrarily complex. Path selection, based on Dijkstra's algorithm, automatically generates intelligent overshooting and backtracking.

An arm-held camera has demonstrated the theory by navigating a simple world: it seeks a landmark based on topology, and adjusts its position based on symmetry; there are essentially no quantitative measures. Because direction-giving is NP-complete, several heuristics were found necessary; one is that the landmark object itself, rather than its views, may be its most compact encoding. Work continues on the related issues of error detection and correction, camouflage, and "self-correcting" directions.

(John Kender, Avraham Lelf, Il-Pyung Park, and David Yang [27, 30, 31, 32]).

3.5 Visual Servoing

A system for tracking moving objects has been demonstrated, and a video of its performance is available. Calibrated (but not registered) stereo cameras image a moving object with an arbitrary trajectory; the PIPE performs an optic-flow computation on the imagery in real-time. Velocity fields are thresholded to find regions of object motion, which are then triangulated to give a 3-D position vector in less than 100 milliseconds. This vector is input to a second order digital filter that compensates for video processing delays, and predicts and smooths the tracking arm's trajectory. Extensions to the system to enable real-time grasping by the hand are in progress.

(Peter Allen, Aleksandar Timcenko, and Billibon Yoshimi [5])

3.6 Integrated Hand-Eye Systems

The Utah-MIT dexterous hand is a four fingered, 16 degree of freedom device of high dexterity, equipped with rich force, position, and tactile sensors. Research continues on autonomous low-level control, grasping kinematics, active sensing for object recognition, tactile sensor design, and the integration of dexterous hands into robotics environments. A working hand-arm system exists that uses a task description language to program grasping and manipulation tasks. It has been used to perform picking-and-placing, handling of liquids, unscrewing fixtures, and active sensing tasks, including vision-assisted grasping.

Robotic analogs of three human haptic sensing strategies have been analyzed and implemented, to recover the 3D shape of objects. Each strategy is inspired, in part, by an object model prevalent in the vision community: polyhedra, superquadrics, and generalized cylinders. These strategies have been complemented by visual input. A real-time linear feature extractor sends line segments to a robust line-based stereo system, which recovers the 3-D axes of surfaces of revolution. These axes are then used by the hand system to orient itself and to explore the object's contour, and to recover the shape of the object.

(Peter Allen, Paul Michelman, and Ken Roberts [1, 2, 3, 4, 6, 7, 8, 33]).

3.7 Sensor-based Volume Registration and Recovery

The PROVER System (Parametric Representation of Volumes: Experimental Recovery) is in prototype, as a testbed to allow exploration of the numerical recovery of parametric representations from multiple types of data, and multiple sensor types. The system explicitly encodes sensor error models.

To guide the implementation, the causes of misregistration in medical volume data have been surveyed. Five major components of registration error have been identified: how features are matched, which similarity metric is computed, which strategy is used to search for correspondence, what image-to-image transformations are permitted, and how the final interpolation occurs. Several different medical sensing systems have been investigated: computerized

tomography, magnetic resonance imagery, positron emission tomography, ultrasound, and single positron emission. A comparative evaluation is in progress.

(Lisa Brown, and Terry Boulton).

3.8 Bayesian Analyses for Image Interpretation Systems

With an eye towards improving the accuracy of photointerpretation tasks, a novel Bayesian recognition strategy is under development. It will efficiently compute likelihoods for object identification, by organizing and then pruning networks of inferences.

(Michelle Baker, and Terry Boulton [9]).

4 References

1. Allen, P.K., and Roberts, K. Haptic Recognition Using a Dextrous Multi-Fingered Hand. Proceedings of the IEEE International Conference on Robotics and Automation, May, 1989, pp. 342-347.
2. Allen, P.K., Michelman, P. and Roberts, K. An Integrated System for Dextrous Manipulation. Proceedings of the IEEE International Conference on Robotics and Automation, May, 1989, pp. 612-617.
3. Allen, P.K. Cooperative Integration of Vision and Touch. Proceedings of the SPIE Conference on Sensor Fusion II. Human and Machine Strategies, Vol. 1198, Nov., 1989.
4. Allen, P. K., and Michelman, P. Acquisition and Interpretation of 3-D Sensor Data from Touch. Proceedings of the IEEE Workshop on Interpretation of 3-D Scenes, Nov., 1989, pp. 33-40.
5. Allen, P.K., Timcenko, A., and Yoshimi, B. Real-Time Visual Servoing, Proceedings of the DARPA Image Understanding Workshop, Sep., 1990. (These proceedings.).
6. Allen, P.K. Issues in Building Intelligent Grasping Systems. Proceedings of the SPIE Conference on Applications of Artificial Intelligence VIII, Apr., 1990, pp. 682-690.
7. Allen, P.K. Mapping Haptic Exploratory Procedures to Multiple Shape Representations, Proceedings of the IEEE International Conference on Robotics and Automation, May, 1990, pp. 1679-1685.
8. Allen, P.K., Michelman, P., and Roberts, K.S. "A System for Programming and Controlling a Multi-Sensor Robotic Hand,". *IEEE Transactions of Systems, Man, and Cybernetics* (Nov./Dec. 1990).
9. Baker, M., and Boulton, T.E. Pruning Bayesian Networks for Efficient Computation. Proceedings of the Sixth Conference on Uncertainty in AI, AAAI, 1990.
10. Boulton, T.E. "Dynamic Digital Distance Maps". *IEEE Journal of Robotics and Automation* (To appear 1990), .
11. Boulton, T.E., and Lerner, M. Energy-based Segmentation of Sparse Depth Surfaces. Proceedings of the IEEE International Conference on Robotics and Automation, May, 1990.
12. Boulton, T E Smooth Surface Reconstruction and Segmentation. Proceedings of the AFA Conference on Curves and Surfaces, Association Francais d'Approximation, Chamnoix, France, Jun., 1990. (Invited mini-symposia paper.).
13. Boulton, T E , and Lerner, M. Energy-Based Segmentation of Very Sparse Range Surfaces. Proceedings of the DARPA Image Understanding Workshop, Sep., 1990. (These proceedings.).
14. Boulton, T E , and Lerner, M Energy-Based Segmentation Proceedings of the SPIE Conference on Computer Vision, Nov., 1990.
15. Brown, L G , and Shvaytser, H "Surface Orientation from Projective Foreshortening of Isotropic Texture Autocorrelation" *IEEE Transactions on Pattern Analysis and Machine Intelligence* (Jun. 1990), .
16. Brown, L.G. "Surface Orientation from Texture Autocorrelation". *Journal of Image and Vision Computing* (In revision 1990).
17. Choi, D.J., and Kender, J.R. "Solving the Depth Interpolation Problem on a Parallel Architecture with Efficient Numerical Methods". *IEEE Transactions on Pattern Analysis and Machine Intelligence* (In revision 1990).
18. Gross, A.D., and Boulton, T.E. Straight Homogeneous Generalized Cylinders. Analysis of Reflectance Properties and a Necessary Condition for Class Membership. Proceedings of the IEEE Conference on Systems, Man, and Cybernetics, Oct , 1989 (Invited paper.).
19. Gross, A.D. Recovery of Straight Homogeneous Generalized Cylinders Using Contour and Intensity Information. Proceedings of the SPIE Conference on Visual Communications and Image Processing 4, Nov., 1989.
20. Gross, A.D., and Boulton, T.E. An Algorithm to Recover Generalized Cylinders from a Single Intensity View. Proceedings of the IEEE International Conference on Robotics and Automation, May, 1990.
21. Gross, A.D. Straight Homogeneous Generalized Cylinders: Constraints from Contour. Proceedings of the DARPA Image Understanding Workshop, Sep , 1990. (These proceedings).
22. Gross, A.D., and Boulton, T.E. Recovery of Generalized Cylinders from a Single Intensity View. Proceedings of the DARPA Image Understanding Workshop, Sep , 1990 (These proceedings).
23. Hatzitheodorou, M. *Shape from Shadows: Theoretical and Computational Aspects*. Ph.D. Th., Department of Computer Science, Columbia University, Dec. 1989.
24. Hatzitheodorou, M. Regularized shape from Shadows. Tech. Rept CUCS-416-89, Department of Computer Science, Columbia University, 1989.
25. Hatzitheodorou, M. A New Illumination Method for Surface Reconstruction Proceedings of the SPIE Symposium on Intelligent Robots and Computer Vision, Nov, 1989.
26. Hatzitheodorou, M. Shape from Narrow Light Beam Projections. Tech Rept. CUCS-006-90, Department of Computer Science, Columbia University, 1990.
27. Kender, J.R., and Leff, A. "Why Direction-Giving is Hard: The Complexity of Linear Navigation by Landmarks" *IEEE Transactions on Systems, Man, and Cybernetics* 19, 6 (Nov./Dec 1989), 1656-1659.
28. Kender, J R Some Issues for the Panel on Methodology and Standards in CVPR Research. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1989.
29. Kender, J R., and Kjeldsen, R. On Seeing Spaghetti: A Transform and a Connectionist Representation for Flexible Extruded Objects Under Range Data. Proceedings of the DARPA Image Understanding Workshop, Sep., 1990. (These proceedings.).
30. Kender, J.R., and Leff, A. Why Direction-Giving is Hard: The Complexity of Linear Navigation by Landmarks. Proceedings of the AAAI Workshop on Qualitative Vision, Aug., 1990.
31. Kender, J R , Park, I.P., and Yang, D. A Formalization and Implementation of Topological Visual Navigation in Two Dimensions. Proceedings of the DARPA Image Understanding Workshop, Sep., 1990. (These proceedings.).
32. Kender, J.R., Park, I.P., and Yang, D. A Formalization and Implementation of Topological Visual Navigation in Two Dimensions. Proceedings of the SPIE Symposium On Advances in Intelligent Systems, Nov., 1990.

33. Michelman, P., and Allen, P. Haptic Perception with a Robot Hand: Requirements and Realization. Proceedings of the NATO ASI conference on Active Perception and Robot Vision, Jul., 1989.
34. Moerdler, M. *Shape from Textures: A Paradigm for Fusing Middle-Level Visual Cues*. Ph.D. Th., Department of Computer Science, Columbia University, May 1989.
35. Roberts, K. Robot Active Touch Exploration: Constraints and Strategies. Proceedings of the IEEE International Conference on Robotics and Automation, May, 1990.
36. Roberts, K. Coordinating a Robot Arm and Multi-Finger Hand Using the Quaternion Representation. Proceedings of the IEEE International Conference on Robotics and Automation, May, 1990.
37. Singh, A. An Information-Fusion Based Approach for Image-Flow Computation. Proceedings of the SPIE Conference on Sensor Fusion II: Human and Machine Strategies, Vol. 1198, Nov., 1989.
38. Singh, A. *Image-Flow Computation: Estimation-Theoretic Framework, Unification and Integration*. Ph.D. Th., Department of Computer Science, Columbia University, May 1990.
39. Singh, A. An Estimation-Theoretic Framework for Image-Flow Computation. Proceedings of the DARPA Image Understanding Workshop, Sep., 1990. (These proceedings.).
40. Singh, A., and Lala, P.K. "A Multilayer Cellular Architecture for a Highly Parallel VLSI Supercomputer". *IEEE Transactions on Computers* (To appear 1990).
41. Soldo, M. Hanvey. Fusion Without Representation. Proceedings of the SPIE Conference on Sensor Fusion II. Human and Machine Strategies, Vol. 1198, Nov., 1989.
42. Soldo, M. Hanvey. Reactive and Preplanned Control in a Mobile Robot. Proceedings of the IEEE International Conference on Robotics and Automation, May, 1990. (Finalist, Best Student Paper competition).
43. Soldo, M. Hanvey. Reactive and Preplanned Control in a Mobile Robot. Proceedings of the DARPA Image Understanding Workshop, Sep., 1990. (These proceedings.).
44. Soldo, M. Hanvey. "Environmental Representations for Mobile Robot Navigation". *ACM Computing Surveys* (To appear 1990).
45. Tarabanis, K., and Tsai, R. Camera Placement Planning Avoiding Occlusion. Test Results Using a Robotic Hand/Eye System. Tech. Rept. CUCS-501-89, Department of Computer Science, Columbia University, 1989.
46. Tarabanis, K., and Tsai, R. Model-Based Planning of Sensor Placement and Optical Settings. Tech. Rept. CUCS-502-89, Department of Computer Science, Columbia University, 1989.
47. Tarabanis, K. Model-Based and Task-Driven Machine Vision Planning. Tech. Rept. CUCS-028-90, Department of Computer Science, Columbia University, 1989.
48. Tarabanis, K., Tsai, R., and Allen, P. Satisfying the Resolution Constraint in the MVP Machine Vision Planning System. Proceedings of the DARPA Image Understanding Workshop, Sep., 1990. (These proceedings.).
49. Tsai, R., and Tarabanis, R. Model-Based Planning of Sensor Placement and Optical Settings. Proceedings of the SPIE Conference on Sensor Fusion II: Human and Machine Strategies, Vol. 1198, Nov., 1989.
50. Weems, C., Brown, C., Kender, J.R., and Poggio, T. "Parallel Vision Algorithms". *IEEE Expert* (Special Issue on Strategic Computing Vision, to appear 1990).
51. Wolberg, G. "A Skeleton-Based Image Warping". *Visual Computer* 5 (1989), 95-108.
52. Wolberg, G., and Boulton, T.E. "Separable Image Warping with Spatial Lookup Tables". *Computer Graphics* 23, 3 (Jul. 1989), 369-378. (Proceedings of SIGGRAPH '89).
53. Wolberg, G. *Separable Image Warping: Implications and Techniques*. Ph.D. Th., Department of Computer Science, Columbia University, June 1990.
54. Wolberg, G. *Digital Image Warping*. IEEE Computer Society Press, Washington, D.C., 1990.
55. Wolberg, G., and Boulton, T.E. A Method and an Apparatus for Separable Image Warping with Spatial Lookup Tables. U.S. Patent Application. Jul., 1989.
56. Wolff, L.B. Shape Understanding from Lambertian Photometric Flow Fields. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June, 1989.
57. Wolff, L.B. Using Polarization To Separate Reflection Components. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June, 1989.
58. Wolff, L.B. Accurate Measurement of Orientation From Stereo Using Line Correspondence. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June, 1989.
59. Wolff, L.B., and Boulton, T.E. Polarization/Radiometric Based Material Classification. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June, 1989.
60. Wolff, L.B., and Boulton, T.E. Experiments For Determining Surface Orientation From Line Correspondence Stereo. Proceedings of the International Joint Conference on Artificial Intelligence, August, 1989.
61. Wolff, L.B. A Photometric Invariant and Shape Constraints at Parabolic Points. Proceedings of the DARPA Image Understanding Workshop, Sep., 1990. (These proceedings.).
62. Wolff, L.B. "Spectral and Polarization Stereo Methods Using a Single Light Source". *International Journal of Computer Vision* (In revision 1990).
63. Wolff, L.B. "Material Classification From the Polarization of Specular Highlights". *IEEE Transactions on Pattern Analysis and Machine Intelligence* (To appear 1990).
64. Wolff, L.B., and Kurlander, D. "Ray Tracing With Polarization Parameters". *Computer Graphics and Applications* (To Appear 1991).
65. Yoshimi, B., and Zaidi, Q. The Effect of Shape, Perimeter and Area on Induced Color Contrast. Proceedings of the ARVO Conference, Mar., 1990, pp. 264.



MIT PROGRESS IN UNDERSTANDING IMAGES

T. Poggio and the staff

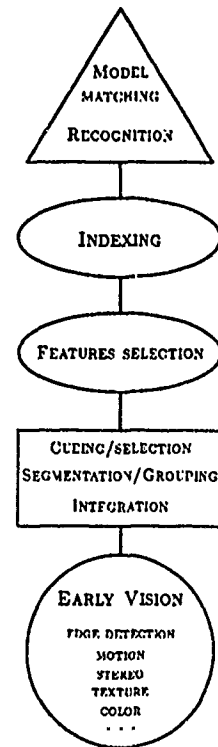
Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139

ABSTRACT

Our program in Image Understanding is now focusing on the critical issues of segmentation, saliency computation and integration of visual cues. These problems need to be solved efficiently in order to exploit in robust systems our ongoing work on object recognition. We have also continued our work on the computation and the use of motion, photogrammetry, analog VLSI circuits and learning.

1 Introduction

Our present approach to vision is reflected in the organization shown in the figure. It consists of several stages which are not strictly sequential (not all forward and backward connections are shown). We feel quite optimistic about the bottom - early vision - and the top levels - model-based recognition - since progress has been and is being made at a significant rate, both in terms of applications and of theoretical foundations. More fundamental work still remains to be done at the intermediate stages, especially at the dual stages of segmentation and grouping and on the problem of feature selection. We have continued our basic research on early vision, though at a lower level of effort than in previous years, and on object recognition along several different directions. We will report on some of those. We have also continued to explore the problem of integration and segmentation using the Markov Random Field model paradigm. At the same time we have begun to attack in new ways the fundamental problems of selection, grouping and again segmentation. In the following, we will sketch some aspects of this overall approach to the vision problem and mention a few of the minor projects.



2 Segmentation and saliency computation

We have described last year our work on grouping and saliency computation. Especially the latter theory has opened a number of interesting areas of research and application. In particular, during the last year Shashua, Spierri and Ullman have extended the approach of finding salient image contours to deal with discontinuity contours, with the goal of using these contours for segregating objects from the background. The general scheme proceeds in two stages. The first is a local estimation of the existence of a discontinuity. For example, at a given location and at a given orientation, we examine whether the points on the two sides of the orientation element are moving in the same or in different directions. The result is a discontinuity map of potential fragments of discontinuity boundaries (motion boundaries in the above example). The next stage pieces together fragments that form long continuous (and preferably closed) bound-

aries. The second stage is similar in nature to the process that is used to find salient image contours based on length and curvature. There are some technical differences that must be introduced in the discontinuity computation. For example, if the saliency computation is applied to this problem without modifications, it often produces more than a single discontinuity contour separating a region from its surround. By incorporating a certain form of local inhibition between neighboring discontinuity boundaries it becomes possible to ensure that only one boundary will survive in such cases. The computation of discontinuity boundaries is being developed by A. Shashua in the context of depth discontinuities, and applied also to motion discontinuities by A. Spoerri.

It is satisfying to note that despite some differences in details, a uniform computation emerges that appears to play a similar role in a number of different processes. The emerging unified scheme is the following. In all of the above computations the first stage produces a measure $M(x, y, \theta)$ that depends on image location (x, y) and the local orientation θ . In the contour saliency computation $M(x, y, \theta)$ is the local saliency (determined e.g. by contrast) of a contour element at (x, y) with orientation θ . In the symmetry computation $M(x, y, \theta)$ is a measure of the local symmetry. In the discontinuity computation $M(x, y, \theta)$ is a measure of the local change (in velocity, disparity, etc.). The second stage takes the map $M(x, y, \theta)$ as input, and produces optimal 1-D contours. The contours are required to be long and smooth, and to maximize the measure $M(x, y, \theta)$ along them.

2.1 Integration in the Vision Machine

As we described last year, the Vision Machine system is our main tool for studying the problem of integrating several visual cues. The Vision Machine consists of a movable two-camera Eye-Head system - the input device - and a 8K CM2. Its parallel early vision algorithms compute edge detection, stereo, motion, texture and surface color in close to real-time. The integration stage is based on the technique of coupled Markov Random Field models, and leads to a cartoon-like map of the discontinuities in the scene, with a partial labeling of the brightness edges in terms of their physical origin. The output of our integration stage feeds a parallel model-based recognition algorithm. We have now developed a new eye-head system which has less degrees of freedom (for instance it does not have zoom lenses) but is much lighter and smaller. We plan to use it as a flexible input device that can be moved to different rooms in the lab or outside it. In addition to representing a focus for our work on the integration of early vision modules and for the development of parallel vision algorithms, the Vision Machine system will also be the testbed for our overall approach to the organization of vision, in which we will integrate and test the different stages and strategies of processing.

3 Object Recognition

In previous reports, we have described a series of approaches to the problem of model-based object recognition,

concentrating on the matching of aspects of object shapes. Recent work has focused on several areas of the domain, based in part on the results of earlier work.

3.1 Formal Analysis of Constrained Tree Search

Earlier reports described the work of Grimson and Lozano-Pérez on the recognition of occluded objects from noisy data, either in the 2D from 2D case, or the 3D from 3D case. The original technique was designed to recognize polyhedral objects from simple measurements of the position and orientation of features in the data, where the features could be edges, vertices, curved arcs, distinctive points along curves, axes of cylinders, patches of surface, etc. The technique searches for consistent matches between object features and data features, using constraints on the relative shapes of pairs of features to reduce the search.

Based on extensive empirical experience, the method was extended to include a Hough transform as a preprocessor, which isolates portions of the search space likely to contain a correct solution and rank orders these subspaces for processing by the recognition engine, and to include the use of a heuristic termination of search, by stopping once an interpretation that is sufficiently strong is found. The addition of these two methods led to a very efficient and robust recognition method.

Although the method has been run on tens of thousands of examples, we have also performed a formal analysis of the approach, in order to understand where its limitations are, and to direct further research efforts. Our analysis considers three different aspects of the problem:

- Selection: Choosing subsets of the data that are likely to come from a single object;
- Indexing: Choosing an object model from the library that is likely to correspond to the selected data subset;
- Correspondence: Finding a legal match, if one exists, between the features of the object model and a subset of the selected data features.

Grimson has established the following results, in some cases in collaboration with Dan Huttenlocher of Cornell:

- If selection is perfect (no spurious data is included) and indexing is correct, then the expected amount of search for constrained tree search methods is quadratic in the number of data and model features.
- If selection is not used and indexing is correct, then the expected amount of search for constrained tree search methods is a combination of a polynomial in the number of data and model features, but is exponential in the size of the correct solution.
- Using the Hough transform to isolate and rank order subspaces of the search space for consideration reduces the values of the parameters in the complexity bounds, but does not reduce the order of the bound from exponential.
- The Hough transform cannot be used to identify solutions to the recognition problem, without also

incurring a significant false positive rate, especially for problems with significant clutter or noise. A similar result holds for another parameter hashing scheme, Geometric Hashing.

- If selection is adequate (where a formal definition of adequate can be given as a function of conditions on the ratio of spurious data to model size), indexing is perfect, and one heuristically terminates the search once a sufficiently good solution is found (where formal methods for defining thresholds for "sufficiently good" are available), then the expected amount of search is quartic in the number of data and model features.
- If indexing returns an incorrect answer (i.e. one tries to match a model not present in the data), even using heuristic search termination, the expected amount of search to deduce that the model is not present in the data is exponential in the problem parameters.

3.2 Selection, or Grouping, Methods

Although these formal results have been developed in the context of constrained search methods, they may have broader implications for other types of recognition. One implication of these results is that effective selection, or grouping, can significantly reduce the complexity of recognition. A second is that efficient indexing methods are necessary if one wants to extend recognition to deal with large libraries of objects. As a consequence, we have focused much of our recent effort on these problems.

One approach to selection is to be model-driven. For example, the Hough transform and Geometric Hashing can be considered as grouping methods, but both require a specific object model, and for large libraries this will be inefficient. A second approach is to be data-driven. Earlier work by David Jacobs on the selection problem had focused on generic data-driven grouping. Jacobs concentrated on methods for finding groups of edge fragments likely to have come from a single object, based on simple measurements on convex sets of edges. He showed that dramatic reductions in search (2-3 orders of magnitude) could be obtained by using these groups as starting points. Moreover, the number of false positive and false negative responses of the recognition system are reduced, since one now concentrates on salient portions of the image first. Jacobs has continued to explore this approach, by examining more general grouping methods for flexible objects and their role in controlling the search explosion.

As a complement to this edge-based approach, David Clemens is developing a region based grouping method. The idea is to use regions of smoothly varying intensity to find groups of edge-based image features that are likely to come from the same 3D object. By interpreting regions and other cues, a relatively small number of large feature groups can be produced without the combinatoric explosion that results from forming all possible groups. Larger groups allow for more efficient matching and model-pose solution. The efficiency of region-based and other feature grouping is currently being compared in a complete recognition system.

As part of this system, a semi-automated 3D model-making program has been developed. Given several images of the model object and rough estimates of the camera pose for each image, the system solves for the 3D model points and for the exact camera poses. Image features are automatically extracted, but feature matches between images must be indicated by the user. This enables us to quickly build 3D models of complicated objects for use in recognition experiments.

Another way of attacking the selection problem is to use multiple sensory cues. Any single visual cue may only weakly define groups of data fragments likely to have come from a single object. Taken in concert, however, several cues may provide much more salient groups. Work by Ed Gamble in the context of the MIT Vision Machine takes one approach to this integration problem, by concentrating on the detection of object discontinuities through integration of multiple cues. Motivated by computational models of human visual attention and eye movement studies, Tanveer Syeda is developing a complementary approach, based on a computational model of the visual attentional phenomenon. Visual attention is that mechanism in brain that allows it to respond selectively to some visual stimulus either in a spontaneous or deliberate fashion. An observer exercising this faculty commonly exhibits two kinds of attentional behavior, namely, the *attracted* attention and the *pay* attention modes. In the former, some aspects of the scene *attract* the unbiased observer's attention, while in the latter, the observer has *a priori* goals in mind when looking at the scene and hence *pays* attention to only those objects/aspects relevant to the goal. In either mode, the end result is a selection of certain aspects of the scene on which to focus the later processing. The purpose behind building a computational model for such a phenomenon is two-fold. From computer vision point of view, such a model can serve as a full-scale feature selection mechanism that can act as a front end for an object recognition system and help identify *interesting* regions in a scene to start the recognition process. Secondly, it can supply a plausible explanation for the mechanism constituting the attentional center in the brain, which studies so far have not revealed (although its presence seems to be well accepted in both psychophysical and physiological literature).

Briefly, the model suggests that the scene represented by the image be processed by a set of interacting feature detectors that generate a hierarchy of maps, representing features such as brightness, color, texture, depth, grouping of edges, and others such as shape, size, symmetry, etc. The feature maps are then processed by filters incorporating strategies for selecting distinctive regions in these maps. The choice of these strategies is guided by a central control mechanism that combines top-down task level and *a priori* information with the bottom-up information derived from the features, to demonstrate either mode of attentional behaviour as desired. Finally, an arbiter module housing another set of strategies selects the most significant features across the feature maps, which can then be used in, say an object recognition system.

In the work done so far, color and texture maps have

been built, and the filters for finding distinctive regions in these maps have been developed. The color map outlines different color regions by not only grouping spatially contiguous regions of similar color, but also labeling the regions with the perceptually seen color. The distinctiveness of a colored region is judged based on its and its neighbors' properties such as the actual color seen, contrast shown, size of the region, etc. The texture map is generated by regarding the image as being generated by a space-limited stationary stochastic process. The segmentation of the textured image is then obtained by a comparison of the AR-spectra of adjacent windowed regions of the image. Properties such as the relative distribution of dark and bright blobs are then made use of to judge the distinctiveness of a region.

Finally, the model supports a parallel implementation of the various feature maps that will make it possible to select distinctive features in a pre-attentive fashion. The success of such a model in object recognition will depend on how effectively it can reduce the combinatorial search involved in the matching stage of recognition.

In related work, David Clemens and David Jacobs have been investigating the ability to order groups of model features by geometric properties that are invariant under projection. Each model feature group would be represented along a surface in a multi-dimensional index space. At recognition time, parameters extracted from each group of image features could be used to address a point in the index space and find only those model groups that could cause the image, thus reducing search significantly. They have derived a lower bound on the dimension of such an index space, and developed a detailed algorithm. Error in feature localization is predicted to have a practical impact on the theoretical performance of the system, but the potential benefits are great enough that indexing remains an attractive approach.

3.3 Transformation Space Sampling

The constrained search approach finds solutions to the recognition problem by searching through a correspondence space, i.e. a space in which individual points correspond to sets of pairings of data and model features. An alternative is to search a transformation space, i.e. a space in which individual points correspond to possible poses of the object. For example, the Hough transform operates by letting each possible pairing of a data feature and a model feature vote in transformation space, then searching for peaks in the result.

Grimson and Huttenlocher have argued that one of the difficulties with using the Hough transform directly as a recognition method is that for realistic sized problems, the likelihood of large peaks in the Hough space occurring at random increases significantly. This is in part because of noise in the sensory data implying that a large volume of transformations must be considered as feasible. This in turn increases the likelihood that several such volumes may intersect at random, leading to a false peak in the Hough space. These false peaks also occur, however, because we use a discrete tessellation of the transformation space to define our Hough space. Thus, each bucket in the tessellated space must integrate together votes from

data-model pairings whose associated possible transformations intersect any part of the bucket, even though those associated transformation volumes may not in fact actually intersect. Todd Cass has devised a very efficient algorithm for removing the second effect on the Hough transform.

The idea is as follows. Suppose we can compute the range of transformations consistent with any pairing of a data feature and a model feature. For example, in the simple case of matching points with an associated orientation, the set of feasible transformations is a disk in the translation subspace of the space, which tracks a helical arc over the range of possible orientations associated with the pairing. Further, suppose that we can easily determine whether a point in transformation space lies within one of these feasible match regions. Then if we sample the transformation space at some regular sampling, we can easily determine the set of feature pairings that are consistent with the pose associated with each sample point, by simply counting the number of match regions that contain that point. If we do the sampling at a fine enough spacing, we can get a close approximation to the Hough transform, using infinitesimal buckets, so that we avoid the integration problem mentioned above. Further, if the sampling is fine enough, the probability that the poses associated with peak values in this sampling are identical with the poses one would find in the continuous pose space approaches one, i.e. the peaks we find this way are likely to be exactly the peaks one would find in the ideal case, and their position in transformation space is likely to be very close to the ideal position.

While sampling the transformation space should perform correctly in the limit of very fine sampling, one cannot guarantee its correctness for arbitrary sample sizes. To overcome this shortcoming, Cass has extended this idea of transformation sampling in the following manner. Consider the volume of consistent transformations associated with a pairing of a data feature and a model feature (for simplicity, consider the case of matching oriented points, so that the volume is a tube with circular cross-section that tracks along a helical arc). Denote the volume associated with the pairing of data feature F_i and model feature f_j by V_{ij} . In essence, by considering all such pairings of features, we are defining a function over the pose space:

$$h(p) = ||\{(i, j) | p \in V_{ij}\}||$$

where p is a point in pose space. This function is piecewise constant, and changes value at the boundary of one of the volumes V_{ij} . Cass argues that by identifying those points in pose space at which the boundaries of two such volumes intersect, one can create a set of sample points with the property that each contiguous region of constant value of h has a sample point in this set. Cass has developed a method for computing all such sample points. One can then repeat the earlier process, at this set of sample points, again using the peak values of h over this set of samples to define solutions to the problem. Both methods lead naturally to efficient parallel algorithms. The first algorithm has been implemented and tested on the Connection Machine, and typically recognizes highly occluded objects in very cluttered scenes in

a few seconds. The second algorithm is currently under development and testing.

3.4 Recognition for Navigation

Besides the work on recognizing objects described above, we have also considered the application of our recognition methods to problems of navigation for mobile robots. David Braunegg, in work described elsewhere in these proceedings, has recently completed a system for automatically building models of world locations and using them for navigation. Since the world changes over time, and the sensory input is imperfect, the system also maintains these models over time as the world changes and as we receive new sensory data which is noisy. The MARVEL system proceeds as follows: To recognize a location, we first take a series of stereo pair images from a single position in the current room. The stereo vision module then finds salient features of the room and abstracts them into the representation which will be used for recognition. Recognition is performed by comparing this representation to room models which were built by the system from similar stereo data obtained previously. The results of this recognition are used to update the existing model to reflect the current state of the room and the importance of the features to the recognition. The system has been tested on nearly 1000 stereo pairs, over a six month period in our laboratory. Its success is reported elsewhere in these proceedings.

4 Using motion

The focus recently in work on motion vision has been on improving accuracy by continuing the computation in time, and by exploiting additional prior information. Good results in motion vision can be obtained using direct motion vision methods, contradicting earlier criticism of the brightness change constraint equation. Direct methods, based on brightness gradients, avoid the complexity of feature extraction and the correspondence problem, also, being *eikonic* computations, they lend themselves to parallel high speed implementation in both analog and digital hardware.

Joachim Heel has developed a number of methods exploiting Kalman filtering, and resampling methods borrowed from computer graphics, that dramatically improve the estimates of depth over those available from just two frames (see paper in these proceedings). His schemes, unlike some earlier ones, do not place special restrictions on the motion, or the arrangement of viewing direction, or make special assumptions about the surfaces being viewed. He has also worked with Satyajit Rao on the intimate integration of early vision modules, such as shape from shading and direct motion vision.

David Michael exploits Kalman filtering in a quite different, more traditional way. In his methods, the Kalman filter is used to update the estimated state of the vehicle carrying the sensor. A model of the vehicle dynamics can be used to reduce the effects of noise in measurements obtained from closely spaced frames. He is also exploring the application of non-linear least-squares techniques to the brightness change constraint

equation over more than just two frames (see paper in this proceedings).

Ali Taalebi is pursuing a direct motion vision method exploiting fixation. If the sensors angular velocity is adjusted so that the image of a particular point in the scene remains stationary in the image, simplification of the constraint equations result that can be exploited by using existing methods developed for the case of pure translation (see paper in these proceedings). Fixation is achieved by servoing the camera rotation actuators to zero out the average optical flow over an image patch centered on the point being fixated.

5 Photogrammetry

Photogrammetry is the science of making measurement by means of images, and so is a field closely allied to machine vision. There are four central problems in photogrammetry: (a) absolute orientation, (b) relative orientation, (c) exterior orientation, and (d) interior orientation. Not so long ago, a closed form solution was found to the least squares version of the problem of absolute orientation, which comes up, for example, in the 'calibration' of a range finding system to be used with a robot arm or a mobile vehicle.

Work now focuses on the problem of relative orientation, of importance in both binocular stereo and long-range motion vision—sometimes referred to as "camera calibration." A new iterative algorithm has been developed by Berthold Horn using unit quaternion notation for representing both rotation and an auxiliary quantity derived from the baseline and the rotation (the auxiliary quantity is actually the rotation in a dual problem). This iterative algorithm for the least squares problem efficiently finds minima of the total error, and starting from a small number of suitably chosen initial rotations locates the global minimum.

In the simpler case, when there are only five corresponding ray pairs in the two camera systems, exact solutions can be found. Because the problem is (highly) non-linear, there are typically a number of different solutions. It has been observed that these solutions almost always come in groups of four. With randomly chosen ray pairs there may be no solutions and sometimes there may be as many as twenty solutions. There have been several attempts recently to prove that there can be at most twenty solutions and also to come up with algorithms guaranteed to find all of them. Berthold Horn has devised an algorithm, using the recently developed concept of m -homogeneous sets of equations, which, starting with twenty roots of a simplified set of equations, tracks these solutions to find the twenty roots of the equations representing the actual ray correspondences. The simplified set of equations can be solved in closed form and the final set of twenty roots may contain complex roots, usually in groups of four. The algorithm in effect constitutes a constructive proof.

6 Analog VLSI circuits for vision

David Standley has successfully implemented an analog chip that accurately and rapidly determines the centroid

and orientation of a bright blob in an image. The 6×8 millimeter chip has an array of 29 by 29 sensors that can determine the centroid position with sub-pixel accuracy. It is extremely fast—the settling time appears to be less than 100 micro-seconds. The chip performs a specialized task that normally requires a great deal of highly repetitive computation if done on a serial computer. Specialized digital systems for moment calculations do exist, but none perform the computation in a single chip.

The chip is based on a method developed by Berthold Horn for turning some area-based computations into contour-based computations. Furthermore, a theorem applicable to certain discrete arrangements of resistors allows one to simplify analog computations by exploiting an equivalence between two apparently quite different uses of a network (The new result is thought to be related to Tellegen's theorem). The upshot of all this is that most of the computation can be done by a simple regular resistive grid connecting the sensing elements.

The work on the centroid-and-orientation chip has laid the basis for more sophisticated analog chips, particularly in the area of motion vision. Ignacio McQuirk has been studying and simulating a family of methods for locating the focus of expansion in order to determine which methods are both trustworthy and conveniently implementable in analog hardware. All of the methods studied are based on direct motion vision methods. Unfortunately, the special techniques used in the position-and-orientation chip do not all transfer to this problem. Also, the computation at each picture cell is more complex, so we anticipate that we may have to depart from the elegant and simple arrangement where the computation is completely unclocked.

7 Learning

As we discussed in the last Proceedings, we have made substantial progress towards a rigorous and powerful theory of learning from examples.

We first explain how to rephrase the problem of learning from examples as a problem of approximating a multivariate function.

To illustrate the connection, let us draw an analogy between learning an input-output mapping and a standard approximation problem, 2-D surface reconstruction from sparse data points. *Learning* simply means collecting the *examples*, i.e., the input coordinates x_i, y_i and the corresponding output values at those locations, the heights of the surface d_i . *Generalization* means estimating d at locations x, y where there are no examples, i.e. no data. This requires interpolating or, more generally, approximating the surface (i.e. the function) between the data points (interpolation is the limit of approximation when there is no noise in the data). In this sense, learning is a problem of *hypersurface reconstruction*.

From this point of view, learning a smooth mapping from examples is clearly ill-posed, in the sense that the information in the data is not sufficient to reconstruct uniquely the mapping in regions where data are not available. In addition, the data are usually noisy. *A priori* assumptions about the mapping are needed to make the problem well-posed. One of the simplest assumptions

is that the mapping is *smooth*: small changes in the inputs cause a small change in the output. Techniques that exploit smoothness constraints in order to transform an ill-posed problem in a well-posed one are well known under the term of *regularization theory*. We have recently shown that the solution to the approximation problem given by regularization theory can be expressed in terms of a class of multilayer networks that we call regularization networks or Hyper Basis Functions. Our main result (Poggio and Girosi, 1989) is that the regularization approach is equivalent to an expansion of the solution in terms of a certain class of functions:

$$f(\mathbf{x}) = \sum_{i=1}^N c_i G(\mathbf{x}; \xi_i) + p(\mathbf{x}) \quad (1)$$

where $G(\mathbf{x})$ is one such function and the coefficients c_i satisfy a linear system of equations that depend on the N "examples", i.e. the data to be approximated. The term $p(\mathbf{x})$ is a polynomial that depends on the smoothness assumptions. In many cases it is convenient to include up to the constant and linear terms. Under relatively broad assumptions, the Green's function G is radial and therefore the approximating function becomes:

$$f(\mathbf{x}) = \sum_{i=1}^N c_i G(\|\mathbf{x} - \xi_i\|^2) + p(\mathbf{x}), \quad (2)$$

which is a sum of radial functions, each with its *center* ξ_i on a distinct data point and of constant and linear terms (from the polynomial, when restricted to be of degree one). The number of radial functions, and corresponding centers, is the same as the number of examples.

Our derivation shows that the type of basis functions depends on the specific *a priori* assumption of smoothness. Depending on it we obtain the Gaussian $G(r) = e^{-(\frac{r}{\sigma})^2}$, the well known "thin plate spline" $G(r) = r^2 \ln r$, and other specific functions, radial and not. As observed by Broomhead and Lowe (1989) in the radial case, a superposition of functions like Eq. 1 is equivalent to a network with one "hidden" layer of units. The interpretation of Eq. 2 is simple: in the 2D case, for instance, the surface is approximated by the superposition of, say, several two dimensional Gaussian distributions, each centered on one of the data points.

The network associated with Eq. 2 can be made more general in terms of the following extension

$$f^*(\mathbf{x}) = \sum_{\alpha=1}^n c_{\alpha} G(\|\mathbf{x} - \mathbf{t}_{\alpha}\|_W^2) + p(\mathbf{x}) \quad (3)$$

where the parameters \mathbf{t}_{α} , that we call "centers", and the coefficients c_{α} are unknown, and are in general much fewer than the data points ($n \leq N$). The norm is a *weighted norm*

$$\|\mathbf{x} - \mathbf{t}_{\alpha}\|_W^2 = (\mathbf{x} - \mathbf{t}_{\alpha})^T W^T W (\mathbf{x} - \mathbf{t}_{\alpha}) \quad (4)$$

where W is an unknown square matrix and the superscript T indicates the transpose. In the simple case of diagonal W the diagonal elements w_i assign a specific weight to each input coordinate, determining in fact the

units of measure and the importance of each feature (the matrix W is especially important in cases in which the input features are of a different type and their relative importance is unknown). Equation 3 can be implemented by the network of Fig. 1. Notice that a sigmoid function at the output may be sometime useful without increasing the complexity of the system (see Poggio and Girosi, 1989). Notice also that there could be more than one set of Green's functions, for instance a set of multiquadrics and a set of Gaussians, each with its own W . Notice that two or more sets of Gaussians, each with its own (diagonal) W , are equivalent to sets of Gaussians with their own σ s.

7.1 The learning equations

Iterative methods of the gradient descent type can be used to find the optimal values of the various sets of parameters, the c_α , the w_i and the t_α , that minimize an error functional on the set of examples. Gradient-descent is probably the simplest approach for attempting to find the solution to this problem, though, of course, it is not guaranteed to converge. We define

$$H[f^*] = H_{c,t,W} = \sum_{i=1}^N (\Delta_i)^2,$$

with

$$\Delta_i \equiv y_i - f^*(x) = y_i - \sum_{\alpha=1}^n c_\alpha G(\|x_i - t_\alpha\|_W^2).$$

In the stochastic gradient descent method the values of c_α , t_α and W that minimize $H[f^*]$ are regarded as the coordinates of the stable fixed point of the following stochastic dynamical system:

$$\dot{c}_\alpha = -\omega \frac{\partial H[f^*]}{\partial c_\alpha} + \eta_\alpha(t), \quad \alpha = 1, \dots, n$$

$$\dot{t}_\alpha = -\omega \frac{\partial H[f^*]}{\partial t_\alpha} + \mu_\alpha(t), \quad \alpha = 1, \dots, n$$

$$\dot{W} = -\omega \frac{\partial H[f^*]}{\partial W} + \Omega(t)$$

where $\eta_\alpha(t)$, $\mu_\alpha(t)$ and $\Omega(t)$ are white noise of zero mean and ω is a parameter. The important quantities – that can be used in more efficient schemes than gradient descent – are:

- for the c_α

$$\frac{\partial H[f^*]}{\partial c_\alpha} = -2 \sum_{i=1}^N \Delta_i G(\|x_i - t_\alpha\|_W^2), \quad (5)$$

- for the centers t_α

$$\frac{\partial H[f^*]}{\partial t_\alpha} = 4c_\alpha \sum_{i=1}^N \Delta_i G'(\|x_i - t_\alpha\|_W^2) W^T W (x_i - t_\alpha) \quad (6)$$

- and for W :

$$\frac{\partial H[f^*]}{\partial W} = -4W \sum_{\alpha=1}^n c_\alpha \sum_{i=1}^N \Delta_i G'(\|x_i - t_\alpha\|_W^2) Q_{i,\alpha}, \quad (7)$$

where $Q_{i,\alpha} = (x_i - t_\alpha)(x_i - t_\alpha)^T$ is a dyadic product and G' is the first derivative of G (for details see Poggio and Girosi, 1990a).

7.2 Interpretation of the network

The interpretation of the HyperBF network is the following. *After learning* the centers of the basis functions are similar to prototypes, since they are points in the multidimensional input space. Each unit computes a (weighted) distance of the inputs from its center, that is a measure of their similarity, and applies to it the radial function. In the case of the Gaussian, a unit will have maximum activity when the new input exactly matches its center. The output of the network is the linear superposition of the activities of all the basis functions in the network, plus direct, weighted connections from the inputs (the linear terms of $p(x)$) and from a constant input (the constant term). Notice that in the limit case of the basis functions approximating delta functions, the system becomes equivalent to a look-up table. *During learning* the weights c are found by minimizing a measure of the error between the network's prediction and each of the examples. At the same time, the centers of the radial functions and the weights in the norm are also updated during learning. Moving the centers is equivalent to modifying the corresponding prototypes and corresponds to task-dependent clustering. Finding the optimal weights W for the norm is equivalent to transforming appropriately, for instance scaling, the input coordinates and corresponds to task-dependent dimensionality reduction.

7.3 Extensions and Applications

Caprile, Girosi and Poggio (1990) have introduced techniques for dealing with two aspects of learning: learning in the presence of unreliable examples and learning from positive and negative examples. These two extensions are interesting also from the point of view of the approximation of multivariate functions. The first extension corresponds to dealing with outliers among the sparse data. The second one corresponds to exploiting information about points or regions in the graph of the function that are forbidden.

From a theoretical point of view, it is also interesting to compare the HyperBF networks with multilayer perceptron schemes. It has been proved that multilayer networks of the perceptron type can approximate arbitrarily well continuous functions. Girosi and Poggio (1989c) prove that networks derived from regularization theory and including Radial Basis Functions have a similar property. From the point of view of approximation theory, however, the property of approximating continuous functions arbitrarily well is not sufficient for characterizing good approximation schemes. More critical is the property of *best approximation*. The main result

of Girosi and Poggio is that multilayer perceptron networks, of the type used in backpropagation, are not best approximation. For regularization networks (in particular Radial Basis Function networks) they prove existence and uniqueness of best approximation.

Regularization networks—of which HyperBFs are the most general and powerful version—represent a general framework for learning smooth mappings that rigorously connects approximation theory, generalized splines and regularization with feedforward multilayer networks. They also contain as special cases the Radial Basis Functions technique (Micchelli, 1986; Powell, 1987; Broomhead and Lowe, 1988) and several well-known algorithms, especially in the pattern recognition literature.

Edelman and Poggio (1990) have applied the technique to the problem of 3D object recognition with promising results. They have been able to synthesize a module that can recognize an object from any viewpoint, after it learns its 3D structure from a small set of 2D perspective views, using the HyperBF network scheme. Their results were obtained so far with simulated wireframe objects and assumed that the problems of feature extraction and matching were already solved. The problems of occlusions and spurious features were ignored. Nevertheless, their results are interesting as a nontrivial application of a technique for learning from examples in which model acquisition is very simple.

From a broader point of view, this application to object recognition can be regarded as just one example of a drastically new approach to computational vision, in which some of the needed modules of a vision system are synthesized (or fine-tuned) from a sufficient set of examples, using a standard machinery, without explicit programming.

8 Reading List

References

- [1] Mario Bertero, Tomaso Poggio, and Vincent Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76:869–889, 1988.
- [2] David J. Braunegg. An alternative to using the 3-D delaunay tessellation for representing freespace. A.I. Memo 1185, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, September 1989.
- [3] David J. Braunegg. Location recognition using stereo vision. A.I. Memo 1186, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1989.
- [4] David J. Braunegg. Stereo feature matching in disparity space. A.I. Memo 1184, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, September 1989.
- [5] T.M. Breuel. Adaptive model base indexing. A.I. Memo 1008, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.
- [6] D.S. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1988.
- [7] Todd A. Cass. A robust implementation of 2d model-based recognition. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, Ann Arbor, Michigan, 1988.
- [8] Todd A. Cass. Robust parallel computation of 2d model-based recognition. In *Proceedings Image Understanding Workshop*, Cambridge, MA, April 1988. Morgan and Kaufman, San Mateo, CA.
- [9] S. Edelman, H. Buelthoff, and D. Weinshall. Stimulus familiarity determines recognition strategy for novel 3D objects. A.I. Memo 1138, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.
- [10] S. Edelman and D. Weinshall. Computational vision: a critical review. A.I. Memo 1158, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1989.
- [11] S. Edelman and D. Weinshall. A self-organizing multiple-view representation of 3D objects. A.I. Memo 1146, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, August 1989.
- [12] E.B. Gamble. A comparison of hardware implementations for low-level vision algorithms. A.I. Memo No. 1173, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, November 1989.
- [13] Davi Geiger and Federico Girosi. Parallel and deterministic algorithms for MRFs: surface reconstruction and integration. A.I. Memo No. 1114, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, May 1989.
- [14] F. Girosi and T. Poggio. Networks and the best approximation property. A.I. Memo 1164, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1989.
- [15] F. Girosi and T. Poggio. Extensions of a theory of networks for approximation and learning: dimensionality reduction and clustering. A.I. Memo 1167, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, March 1990.
- [16] F. Girosi, T. Poggio, and B. Caprile. Extensions of a theory of networks for approximation and learning: outliers and negative examples. A.I. Memo 1220, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, May 1990.
- [17] W. Eric L. Grimson. The combinatorics of heuristic search termination for object recognition in cluttered environments. In *Proc. First Europ. Conf. Computer Vision*, pages 552–556, 1990.
- [18] W. Eric L. Grimson. The combinatorics of object recognition in cluttered environments using constrained search. *Artificial Intelligence*, 1990.

- [19] W. Eric L. Grimson and Daniel P. Huttenlocher. On the sensitivity of the hough transform for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:255-274, 1990.
- [20] W. Eric L. Grimson and Daniel P. Huttenlocher. On the verification of hypothesized matches in model-based recognition. In *Proc. First Europ. Conf. Computer Vision*, pages 489-498, 1990.
- [21] W. Eric L. Grimson and Tomas Lozano-Perez. Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research*, 3(3):3-35, 1984.
- [22] W. Eric L. Grimson and Tomas Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(4):469-482, July 1987.
- [23] W.E.L. Grimson. On the recognition of curved objects in two dimensions. A.I. Memo 983, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.
- [24] David W. Jacobs. The use of grouping in visual object recognition. A.I. Technical Report No. 1023, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1988.
- [25] C. A. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constr. Approx.*, 2:11-22, 1986.
- [26] S. Omohundro. Efficient algorithms with neural network behaviour. *Complex Systems*, 1:273, 1987.
- [27] T. Poggio and S. Edelman. A network that learns to recognize three-dimensional objects. *Nature*, 343:263-266, 1990.
- [28] T. Poggio and F. Girosi. A theory of networks for approximation and learning. A.I. Memo No. 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.
- [29] T. Poggio and F. Girosi. HyperBF: a powerful approximation technique for learning. *Network*, (to appear), 1990.
- [30] T. Poggio and the staff. MIT progress in understanding images. In *Proceedings of the DARPA Image Understanding Workshop*, Cambridge, MA, April 1988. Morgan Kaufmann, San Mateo, CA.
- [31] T. Poggio and the staff. M.I.T. progress in understanding images. In *Proceedings of the DARPA Image Understanding Workshop*, pages 56-74, Palo Alto, CA, May 1989. Morgan Kaufmann, San Mateo, CA.
- [32] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317:314-319, 1985b.
- [33] M. J. D. Powell. Radial basis functions for multi-variable interpolation: a review. In J. C. Mason and M. G. Cox, editors, *Algorithms for Approximation*. Clarendon Press, Oxford, 1987.
- [34] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. W.H.Winston, Washington, D.C., 1977.
- [35] S. Ullman and R. Basri. Recognition by linear combinations of models. A.I. Memo No. 1152, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.
- [36] D. Weinshall. Direct computation of 3D shape and motion invariants. A.I. Memo 1131, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, May 1989.

Maryland Progress in Image Understanding

John (Yiannis) Aloimonos Larry S. Davis Azriel Rosenfeld
Computer Vision Laboratory, Center for Automation Research,
University of Maryland, College Park, MD 20742-3411

ABSTRACT

Research in the Computer Vision Laboratory at Maryland is focused on problems whose solutions would constitute significant progress in image understanding. This paper describes some of the fundamental problems that limit the performance of vision systems, and indicates how our research is addressing these problems. The paper is organized around some of the answers to the question "Why is vision hard?"

1 Why Vision Is Hard: Visual Problems Are Ill-Posed

There exist two general goals for practical vision systems: navigation in a complex environment and recognition of classes of objects (such as people or trees) in a complex scene. A large proportion of the research on computer vision addresses one of these two goals, explicitly or implicitly. But achieving these goals presents great difficulties.

These difficulties were realized during the 1960s and '70s after the failure of early attempts to build complete vision systems, i.e. systems that used knowledge at all levels including domain-specific information. "In order to complete the construction of such systems it is almost inevitable that corners be cut and overly simplified assumptions be made" [Brady, 1982]. Doing this results in a system capable of performing a limited set of tasks, but does not enhance our general understanding of vision.

At about that time it was proposed [Marr, 1982] that many visual tasks depended on solving the following problem: From one or more images of a scene, derive an accurate three-dimensional geometric description of the scene and quantitatively recover the properties of the objects in the scene that are relevant to the given task. If we can recover an accurate description of our environment, we can navigate and avoid obstacles, and if we can accurately recover the properties of an object (shape, reflectance, color, etc.) we can use them to recognize it.

How can this recovery be accomplished in a complex visual environment? By following the general

principles for the design of complex systems [Feldman, 1985]. We divide the visual system into functional components, thus breaking the overall task into autonomous parts, and analyze these components individually. We then define the representation of information used by the components and the language of communication among them. The components are then tested individually, in pairs, and all together.

In a visual system, according to the paradigm set forth by Marr, the components are subsystems that recover specific properties of the scene from images. We call these subsystems *modules*. The majority of computer vision research has been devoted to the study of such modules and their integration. The study of human and animal perception provides evidence as to the nature of the modules. For example, one source of evidence for the existence of modules in the human visual system is the study of patients with disabilities that come from brain lesions. Another source is the experiments performed by psychophysicists in which a particular module of the human visual system is "isolated"; examples are Julesz's [1971] experiment on stereoscopic fusion without monocular cues, Land's [1971] demonstration of the computation of lightness, Gibson's [1950] experiments on the perception of shape from texture, etc. Such studies suggest that cues such as shading (image intensity variation), texture (distribution of surface markings), contours and outlines (image discontinuities), color, motion and stereo are very helpful in recovering properties of the scene from images. In computational vision, names have been given to many of these modules: Shape from shading, shape from texture, shape from contour, shape and depth from stereo, structure from motion, direction of light source from intensity, physical discontinuities from intensity discontinuities, motion from image intensity derivatives, etc.

However, during the image formation process the three-dimensional world is mapped into two dimensions, and one dimension is lost. This creates many problems when we try to solve the inverse problem of recovering the world from the image. A problem is ill-posed [Hadamard, 1923] if its solution either does not exist, or is not unique, or does not depend continuously on the data. Poggio and his colleagues [Poggio et al., 1985] realized in the early 1980's that most early vision problems are ill-posed. This ill-posedness is one of the reasons why vision is hard; the next section describes our contribution to this general problem of

The support of the Defense Advanced Research Projects Agency (ARPA Order Nos. 6350 and 6989) and the U.S. Army Engineer Topographic Laboratories under Contracts DACA76-88-C 0008 and DACA76-89-C-0019 is gratefully acknowledged.

visual recovery. Our work here has led to some interesting new mathematics of recovery.

2 Ill-Posedness: Boundary Preserving Regularization, Integration, and Active Vision

2.1 Boundary preserving regularization

This part of our work studies the general recovery problem, i.e. how to recover an unknown function of the scene from an image (or images), when the available constraints are not enough (the function can be depth, shape, albedo, optic flow, etc.).

Poggio et al. [1984] discusses the application of regularization to low-level vision. Write $L = 0$ for the constraint relating the image data to the unknown (e.g. $L = I_x u + I_y v + I_t$, for the computation of optic flow (u, v) from an image $I(x, y, t)$; $L = I - \tilde{I}$ for the case of image reconstruction, where \tilde{I} is the observed intensity) and $S = 0$ for the smoothness constraint.

For the problem of interpolating an intensity function, reasonable choices for S are the first and second derivatives of intensity [Poggio et al., 1984], or some linear combination of the two derivatives. For the two dimensional flow problem, Horn and Schunck [1981] suggest using several first derivative constraints:

$$S_1 = \frac{\partial v}{\partial x}, S_2 = \frac{\partial v}{\partial y}, S_3 = \frac{\partial u}{\partial x}, S_4 = \frac{\partial u}{\partial y}.$$

Another option is to use second derivative constraints. Still another two-dimensional constraint, if the unknown is a scalar, is the derivative of the unknown in the gradient direction. We do not expect these constraints to be exactly zero any more than we would expect L to equal zero exactly.

There are three standard ways to balance data consistency and smoothness requirements. One is to minimize $\sum L^2$ subject to the constraint that $\int \sum S_i^2 \leq \text{Max}$; the second is to minimize $\int \sum S_i^2$ subject to $\sum L^2 \leq \text{Max}$; and the third is to minimize $\sum L^2 + \lambda \int \sum S_i^2$. Here Max is an upper bound on the amount of permissible smoothness (or data consistency) constraint error; the sums are over all data points, the integral is over all space, and $\lambda > 0$ is a parameter to be determined. Max and λ control the relative importance of smoothness and consistency with the image data.

We work with the third kind of regularization that uses the parameter λ . The Euler-Lagrange equations we have to solve are linear in L and S , assuming L is linear. If L and S are linear in the unknown, solving these equations is easy. Provided we pick a reasonable λ the quality of the solution is fairly good except at discontinuities. Near discontinuities the solution is much smoother than it should be. If we use a second derivative smoothness term, we get oscillations in the solution not present in the data or the real world. That is because we have made the first derivative too smooth; there should be a big jump in the first

derivative in the vicinity of the discontinuities. If we reduce the value of λ , we under-smooth and increase noise vulnerability in regions where the unknown is smooth. We might want to let λ vary from position to position, but there is enough difficulty finding one good average λ for the whole image. We could choose to apply regularization only over regions where the unknown is smooth. But we cannot know a priori where the boundaries of such regions are. We can apply any iterative regularization procedure: regularize, segment, regularize, segment, etc. But the initial regularization loses valuable information about discontinuity location because it smooths over discontinuities.

It is clear that this is a very important problem, closely connected to the problem of segmentation.¹ The problem is hard, from a mathematical viewpoint, because the function to be recovered is usually not "nice" (for example it has discontinuous derivatives, i.e. corners) and there is noise in the input data (image). Trying to reconstruct a function with corners (discontinuities), we can follow one of the following two general approaches: (a) to make a rigid distinction between discontinuity and non-discontinuity points, or (b) to accept that there are many kinds of discontinuities (some rounder, some sharper) and reconstruct the function while smoothing as little as possible over discontinuities. Consider the following illustrative example. If we have to recover a function like the one in Figure 1, the first approach will attempt to find the corner point A and then reconstruct by regularizing between discontinuities (Figure 2), while the second approach will reconstruct while smoothing as little as possible in the corner (Figure 3). In intuitive terms, the first approach "believes" in segmentation, while the second one doesn't.

Our contribution here follows the second approach. We have developed a "convex" theory of discontinuous regularization and a "linear" theory [Shulman and Aloimonos, 1988a; Aloimonos and Shulman, 1987; Shulman and Hervé, 1990; Shulman and Aloimonos, 1988b]. A quadratic smoothness measure implicitly assumes S is Gaussian and thus over-penalizes large values of S . Discontinuities do occur and very large values of S are much more likely than a Gaussian distribution would allow. For small values of S , a Gaussian distribution seems to be a reasonable approximation. It is only at points where there is a sharp jump in the unknown that we need to apply a non-quadratic smoothness penalty. Thus we replace the condition

$$\text{minimize } \sum L^2 + \lambda \int \sum S_i^2$$

$$\text{by: } \text{minimize } \sum L^2 + \lambda \int g_{T_i}(S_i)$$

where $g_{T_i}(S_i) = S_i^2$ if $S_i \leq T_i$. Here T_i is a threshold

¹The importance of the problem was also signified by the recent AFOSR Workshop on the "Encounter of Computer Vision and Mathematics", organized by Profs. R. Bajcsy and P. Lax of the University of Pennsylvania, in May 1990.



Figure 1.



Figure 2.



Figure 3.

to be determined. The problem is : what should g_T be for large S_i ?

One possible answer to this query is $g_T(S_i) = T_i^2$ if $S_i > T_i$. This has been proposed in different notation by Blake and Zisserman [1987], Marroquin [1985], Mumford and Shah [1985], and Geman and Geman [1984]. The intuition is that there are two kinds of points: discontinuities and non-discontinuities. At non-discontinuities, S is Gaussian. At discontinuities, all values of S are equally likely. Aside from any computational problems with the variational condition, the idea that all large values of S are equally bad is questionable. We have computed many histograms of differences of intensity; the tails are not particularly flat. In a crowded room, depth distances between occluding objects are not uniformly distributed. Very large distances are unlikely because there is bound to be some other object between two objects that are very far apart. The equal goodness assumption is saying, in the case of a second derivative smoothness constraint on depth, that the world consists (roughly speaking) of planar surfaces and sharp corners (discontinuities in orientation). Indoors, this assumption is often true, but an outdoor scene often has more rounded than sharp corners.

Actually, we simply do not know what the best penalty function is, i.e. what g_T should be used. We do not know the probability distributions of the S_i .

Let us assume that

- (1) The distribution is symmetric about 0; so, for any x , it is equally likely that $S_i = x$ and that $S_i = -x$.
- (2) The distribution is a mixture, so we can write $S = (1 - B)G + B \cdot H$ where B is a binary random variable taking the values 0 and 1 (not bad point; bad point), G is a Gaussian random variable, and H is a random variable with unknown probability density.

The best we can do is choose a penalty function P that minimizes the expected solution error under the worst possible distribution H . Thus we want a P such that $\max_H ESE(P, H) = \min_p \max_H ESE(p, H)$.

Looking for the minimax penalty function is the same as finding the worst, least informative H , and choosing the P corresponding to that H . This least informative H is probably too uninformative. We assumed only that H is symmetric; otherwise it is entirely arbitrary. In practice we can learn additional constraints on H that are reasonably certain to hold. Furthermore we are assuming we know the expected value of B , i.e. we know the expected fraction of bad points. This is equivalent to knowing the threshold, T . In reality, finding a good T is nontrivial.

Huber [1981] shows that the least favorable distribution, the one causing the greatest expected mean square error, is the distribution corresponding to the penalty function

$$g_T(x) = x^2 \text{ for } x \leq T,$$

$$= T^2 + 2T|x-T| \text{ for } x \geq T.$$

This function is convex; thus all local minima are global minima. If we add a small quadratic to g_T , we obtain a strictly convex function

$$g_{T,\epsilon}(x) = x^2 \text{ for } x \leq T$$

$$= T^2 + 2T|x-T| + \epsilon(x-T)^2 \text{ for } x \geq T.$$

Now we are guaranteed unique local minima to our variational condition provided we have enough data points. In practice, we do not seem to need the extra term involving ϵ . If we use this expression, ϵ can be any small positive number.

Figure 4 is an image of a complex scene. It was photographed in two views, with camera distance chosen such that the optical flow values should always be of the order of magnitude of one pixel and the optical flow equation, which assumes short-distance motion, is valid. Unfortunately the scene contains many specular points and few curved objects and thus is not ideal for our algorithm, but the preliminary results we get are fairly good anyway; in fact our detection of depth discontinuities is quite impressive. We only have to compute the horizontal component of the flow; we display it as an intensity map in Figure 5 [Shulman and Hervé, 1990].



Figure 4.



Figure 5.

Our second contribution is the linear theory of discontinuous regularization [Shulman and Aloimonos, 1988b; Shulman, in preparation]. Standard regularization minimizes the functional

$$\int L^2 + \lambda S^2$$

where L is the constraint and S is smoothness; we instead minimize

$$\int \sum_0^\infty a_i (D^i L)^2 + b_i (D^i S)^2$$

where a_i , b_i are parameters to be determined. There exist many justifications for why we use this expression; they have been presented in [Aloimonos and Shulman, 1987]. There are various ways for obtaining the parameters, most of which are intractable. We have implemented an adaptive estimation of the coefficients using data-dependent learning from examples [Aloimonos and Shulman, 1987], and we have applied it to the problem of 1-D interpolation. The results were up to 80% better than ordinary regularization (in the sense of mean-square error). Shulman applied the theory to the problems of image restoration and optic

flow computation. In the restoration application, standard regularization and discontinuous regularization were used to reconstruct an image to which correlated, uniform noise had been added. The latter result had a 4% lower mean-square error. The optical flow application used a planar dot pattern whose motion had a discontinuity along the diagonal. When the flow was reconstructed using standard and discontinuous regularization, the latter reconstruction had a 19% lower mean-square error. Applications of the theory to other fields are discussed in [Shulman and Aloimonos, 1988b].²

A class of filters for image smoothing and differentiation have been developed using a combination of regularization techniques and local methods. The fundamental problem of smoothing and differentiating of noisy images has been previously approached in two different ways: 1) Minimization of a smoothness functional, a theoretically well understood procedure but one that involves the solution of a very large system of equations involving all the pixels of the image, for each image. 2) Use of small scale, ready made filters for local smoothing. A method has been developed [Weiss, 1989; Meer and Weiss, 1989] that combines the advantages of the two approaches. General filters are constructed for local windows of the image, derived from maximization of smoothness or from "regularization" theory. In this way the theoretically robust minimization process becomes suitable for practical implementation, possible in real time, and is readily adaptable to local image properties. Filters for more reliable derivatives have also been derived [Weiss, 1989].

2.2 Integration

Within the recovery paradigm, it has become clear that individual visual computations are underconstrained, but additional constraints can be introduced by using multiple sensors or by combining multiple techniques for inferring scene information from images. We have developed [Aloimonos and Shulman, 1989] general methods for integrating visual modules and we are currently performing experiments using these methods. We have studied the coupling of stereo and motion [Aloimonos and Hervé, in press], motion and texture [Aloimonos, 1989], motion and multiple views [Basu and Aloimonos, 1987], contour and motion [Aloimonos and Shulman, 1989], and motion from binocular flows without correspondence [Duncan and Li, 1989]. Finally, we have demonstrated that when cues conflict, humans experience visual illusions [Aloimonos and Huang, 1990]. For example, when humans observe a set of points lying on the surface of a rotating cylinder through a circular aperture, they perceive a rotating sphere. We have developed

²For example, Yuille and Grzywacz [1988] successfully applied this theory to the problem of motion coherence.

computational models explaining these illusions that are consistent with results reported in the perception literature.

2.3 Active vision

An important method for introducing additional constraints into visual computations is to control the parameters of the sensor, for example its spectral sensitivity, its focal length or its position and orientation. This approach is known as the "active vision" paradigm.

We pointed out several years ago that if the observer is active many recovery problems become well posed and some of them become linear. Since then we have concentrated on the study of specific activities. We showed [Aloimonos, 1989] how an active observer can understand shape by unifying shading and texture in a simple manner. We have also started some work on exploratory vision, i.e. searching through the space of activities in order to find the optimal one (i.e. the one that will result in the most robust reconstruction) [Hervé and Aloimonos, 1990]. Most of our recent work in active vision falls in the paradigm of purposive and qualitative vision and will be described later.

3 Why Vision Is Hard: The Visual World Is Noisy

Even well posed (or regularized) visual computations are often numerically unstable, if noise is present in both the scene and the image. Scenes are usually corrupted by "noise" coming from various sources (dust, fog, sun glitter, etc.). The image formation process introduces additional noise. As a result, many problems which theoretically have unique solutions become very unstable in the presence of input noise. To make the exposition simpler we concentrate on the problem of visual motion interpretation. Our work here can be classified into two broad categories: (a) geometric and statistical analysis of the problem in order to construct provably optimal estimators and to understand their inherent limitations; (b) the paradigm of purposive and qualitative active vision.

3.1 Optimal visual motion algorithms

We are now in the third phase of research in visual motion. The first phase was concerned about what can be inferred from dynamic imagery, i.e. how many features in how many views are needed in order to guarantee uniqueness. The second phase was devoted to extracting closed-form solutions for structure and 3-D motion given retinal correspondence of points, lines [Spetsakis and Aloimonos, in press], or other features [Spetsakis and Aloimonos, 1989].

The third phase is devoted to the quest for robust algorithms. It has become very clear in the past few years that the problem of estimating structure and 3-D motion from dynamic imagery is unstable in the presence of noise. In [Spetsakis and Aloimonos, 1988] it was shown that no matter how many point correspondences we use in two frames, we cannot reduce the

error in the computation of structure using any non-linear optimization technique (such as the ones existing in the literature). A simple intuitive geometric reason for this is the following (see Figure 6).

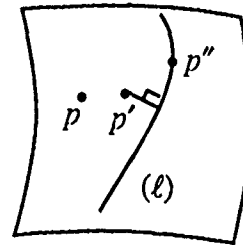


Figure 6.

Let p be a point before the motion, and suppose we correspond it (wrongly) with point p' . The motion constraints require that the true corresponding point p'' lie on a line (l) which is parameterized by the 3-D motion and structure. Any algorithm that attempts to find a robust solution must try to minimize the input error $p'p''$. However, we can only minimize one component of this vector, the distance of p' from l . Thus computation of structure from two frames in the presence of noise is at the mercy of input noise.

This argument doesn't apply to the computation of motion parameters. We have developed [Spetsakis and Aloimonos, 1988b] an optimal algorithm for computing 3-D motion from two frames under the assumption of Gaussian noise. The algorithm is optimal in the maximum likelihood sense and results in a weighted least-squares approach. After realizing that the optimal approach, in the presence of a 1% error in the input,³ could result in about a 50% error in the output, we employed redundancy, i.e. we used multiple dynamic frames, and the results improved considerably [Spetsakis and Aloimonos, 1988c]. Using more frames increases the robustness because of the additional constraints. The following diagram (Figure 7) displays the behavior of the above algorithms in the presence of noise, along with the behavior of one more algorithm that was replicated from the literature. It is clear that these techniques cannot yet be used by machines. The horizontal axis denotes error (noise) in the input and the vertical one denotes error in translation or rotation.

Recently, in our effort to address the stability of structure from motion (SFM), we unified [Spetsakis and Aloimonos, in press] the treatments of SFM with regard to the input used. We introduced a new statistical definition of feature points under which point features and line features are just the two extremes of a spectrum of possible features. Almost any pixel in

³1% in units of focal length, which for the focal lengths of commercially available cameras corresponds to about a 4-6 pixel error in the image displacement.

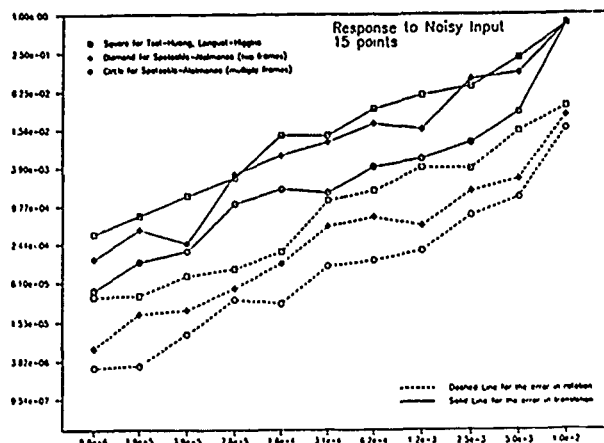


Figure 7.

the image can be classified and used as a feature point in this scheme. Based on this definition we have designed an optimal algorithm for the SFM problem that can utilize information from the whole image. The input to the algorithm is the image displacement, and its uncertainty, at each pixel for a set of three frames. The only assumptions used are rigidity and Gaussian noise in the image displacements. The outputs are the parameters of the motion between the frames and the structure of the scene.

The theory behind this approach is simple and elegant; it can be extended in several ways (e.g. to multiple frames); and it was developed with noise stability in mind. More importantly, the new statistical definition of the features relaxes the requirements on the image displacement computation. In fact, if the tangential component of a displacement cannot be computed then its uncertainty is set to infinity; the algorithm can tolerate infinite uncertainty for all the tangential components. In this way the aperture problem is avoided.

We have also developed analytical techniques for analyzing the numerical errors introduced by the process of discretization [Kamgar-Parsi et al., 1989a; Brosh et al., 1989]. These techniques have been applied to various structure from motion algorithms, with discouraging results as far as the robustness of these algorithms is concerned. Recently we have been working on a new approach [Jasinschi, 1989a; Jasinschi, 1989b] to representing uncertainty in low-level vision. If, for example, we take into account the constraints associated with motion uncertainty, we are able to devise an adaptive procedure for estimating the various parameters involved in space-time filtering.

Finally, we have begun to apply robust statistics to various image estimation problems. As one example, representation of an image by piecewise polynomial surfaces may be of importance for some tasks. In [Meer et al., 1989] a new algorithm was introduced which recovers the fit corresponding to the absolute

majority of the pixels in the processing window. The algorithm uses the least median of squares estimator (*LMedS*) in a two-stage procedure. The goal of this work has been to understand the benefits and difficulties of applying the *LMedS* estimator to computer vision problems.

3.2 Purposive and qualitative active vision

There is a disconcerting lack of visual systems which perform well in real-world environments, particularly when compared to the amount of mathematical theory published on the subject. There seem to be several reasons for this.

One reason is that extracting useful visual information from images probably involves a very large amount of computation. The visual cortexes of animals that perform complex visually moderated behaviors contain millions of neurons, each of which performs computations which require thousands of computer steps per second to simulate, and possibly many more. Much of this capacity is probably necessary to carry out whatever cortical image processing occurs.

A somewhat related reason is the perception that practical results will eventually flow from a successful theory rather than vice versa.⁴ This probably has more to do with the lack of any practical systems to work with than with philosophical conviction, since historically, empirical engineering applications or unexplained observations have preceded theoretical developments at least as frequently as the reverse. If there were suddenly to appear a number of machine vision systems working robustly in different real-world domains, it is quite probable that theories explaining their commonality would soon appear.

There is a third reason that may explain the dearth of examples of working vision systems, which is that the generally accepted goals for such systems may be misplaced, or at least over-ambitious. The two commonly held touchstones for practical vision systems, recognition and navigation, are high-level objectives. If both were achieved, automatic systems would have many of the capabilities of the human visual system.

Another problematic aspect of the recovery (or reconstruction) school of thought is the fact that visual computations involve finding the value of some real quantity, and usually the success of the visual task relies on the accuracy of the first or second decimal digit of that quantity. As a result, most machine visual tasks are unstable. A slight error in the input is enough to destroy some computations. How can we perform robust visual computations that can be reliably used for accomplishing various tasks?

If we could solve the general recovery problem we would be able to perform many visual tasks, but luckily, it is not always necessary to perform general

⁴This point of view was suggested by Nelson [Nelson and Aloimonos, 1988].

recovery. Rather, we can consider more specific problems! We need vision in order to accomplish tasks that are essential for our survival (recognize friends, enemies, food, avoid danger, etc.). But to carry out a specific task, we do not need to completely recover the world and its properties. When we want to move across a crowded room, we just need to avoid obstacles; it is not necessary to reconstruct the scene and thus know that the person in the corner is smiling! Clearly, if we could reconstruct the scene our task would be very simple; but it is obvious that complete reconstruction is not necessary.

We can study visual abilities in a purposive manner, keeping in mind a basic question: What am I going to use this visual ability for? What tasks can be performed using it? If we study vision in a purposive, utilitarian [Ramachandran, 1989], or animate [Ballard, 1989] manner, the problems that we formulate are much simpler, since they are relevant to the task at hand. Since they are simpler, they can be solved by qualitative techniques that exhibit robustness properties.

Although the foundations of purposive and qualitative vision lie in mathematical and engineering considerations, it seems to be consistent with evolution.⁵ Accepting that the ultimate goal of an organism is survival, visual abilities should have evolved in such a way that they served survival purposes. Thus, visual abilities for avoiding danger, recognizing food, recognizing mates, friends and enemies developed. But although some of these abilities were based on common principles (for example the ability to intercept a moving object and the ability to avoid a moving object are both based on the structure from motion module), they were possibly developed at different times and it is probable that they are implemented by separate hardware. From this point of view we may expect that the machinery of the brain devoted to vision consists of various independent processes (which of course communicate) that are devoted to the solution of specific visual tasks. That seems to be also the view of leading neuroscientists [Regan].

Consider a general⁶ vision system of the future, as we envision it based on our current understanding of visual recovery. That system will consist of a large number of modules, each of which will be devoted to recovering a property of the world from a series of images. There will be modules for shape from shading [Horn, 1986], shape from texture [Aloimonos, 1988], structure from motion [Ullman, 1979], etc. All these modules will communicate and cooperate in building an accurate description of the environment (i.e. reconstructing it). Of course, the system will also have high-level modules which, using the outputs of the other modules, will perform planning and reasoning.

⁵The philosophical standpoint is presented by Searle in his book on intentionality [Searle, 1984].

⁶By general we mean capable of performing a nontrivial number of visual tasks, but not necessarily as good as human vision.

Here, we only consider modules interacting with the image or with immediate representations of it.

What will this general vision system do? We will expect it to be able to carry out certain basic tasks. We expect it to be able to move around in its environment and to understand visual motion. Thus it should be able to perform kinetic stabilization (passive navigation), i.e. to understand its own motion from images and adjust it accordingly. It should also be able to avoid obstacles. It should be able to avoid moving objects that are on a collision course with it. It should be able to detect moving objects in its surroundings and track them and observe them. It should be able to intercept a moving object (prey catching), or to extend its arm and catch a small moving object. It should be able to coordinate its hands and eyes to pick up things. It should be able to solve visual rendezvous problems, such as putting a stick through a hole; and so on.

It is clear that most of these tasks are simple applications of the structure from motion or passive navigation module. Indeed, suppose that the system has a robust structure from motion module, i.e. a module that takes as input a sequence of images and gives as output the structure of the imaged scene as well as the relevant 3-D motion parameters.⁷ Then the system can detect its own motion and can reconstruct the scene, which allows it to avoid obstacles and to solve visual rendezvous and hand-eye coordination problems. It can determine the 3-D motion of a moving object, estimate its trajectory, and thus avoid something that is going to hit it, or catch something by positioning its arm at a specific point. Having a robust structure from motion module is thus very powerful, as it can solve all the above-mentioned tasks. It is no wonder then that this module has attracted so much attention in the past 15 years and has created a very rich literature. (See [Bandopadhyay, 1986] for a review.) However, despite the numerous mathematically sophisticated and elegant theories that have been presented, no one has demonstrated important practical applications yet.

We have two alternatives. The first is to continue our research on recovery, to try to understand why existing approaches are unstable, to develop provably optimal estimators of structure from motion, and to introduce noise remedies such as redundancy. The hope here is that our work will result in the best possible structure from motion module and that this module will be good enough (i.e. robust); or we might be surprised to discover that the best is not good enough for some tasks.

The second alternative is to reconsider our viewpoint about the recovery paradigm, and work "around" the problem. This alternative suggests that

⁷If the system is moving in a static scene, then the structure of the scene is recovered along with the 3-D motion of the system. If the system is stationary, then the shape and 3-D motion of moving objects are recovered. When both the system and parts of the scene are moving, only their relative motion can be recovered.

we should not try to solve the abstract structure from motion problem by developing the structure from motion module. Instead, it suggests that we must ask the question: What tasks will I perform if I have a structure from motion module? After the tasks have been identified, we should solve them directly and not as an application of a general module. For example, we should directly solve the problem of avoiding obstacles. We can ask: Is this moving object coming closer to me? If so, where is the focus of expansion (FOE)? Is it inside the boundaries of the image or outside? If it is inside, does this mean that the moving object will hit me? If it is going to hit me, how long will it take with respect to my reaction time?

Can we solve such a collection of problems in a robust manner? If we can solve such problems directly, the general structure from motion module will no longer be needed.⁸ Moreover, because we are now asking simple questions that have small numbers of possible answers, the potential exists that we will be able to achieve robust solutions, since the solutions are qualitative.

We thus see a new paradigm emerging: that of purposive and qualitative vision (which should of course be active). In this framework, one does not regard a vision system as a collection of modules whose purpose is to reconstruct the world and its properties and thus provide information for accomplishing various tasks. Instead, one regards a vision system as a collection of processes, each of which solves (or groups of which solve) a particular visual task. If we look at computer vision in this way, we are no longer considering vision in isolation, as the recovery school of thought does, but as a part of a larger process in which vision is used as a front end. We are currently designing Medusa, a simple, robust qualitative machine that can perform many navigational tasks in real time; it is described in another paper in these Proceedings.

4 Why Vision Is Hard: Visual Objects Are Hard to Define

Even if we have succeeded in reconstructing the world, in order to recognize objects we need to compare our reconstruction with models of candidate objects. But how do we model a bush or a chair? Researchers have proposed object modeling techniques that can generate a large variety of objects. It is not obvious, however, how to use these techniques to capture the variability of natural objects, or the great variety of artificial objects that can all belong to the same class. Existing machine vision systems have not attempted to handle

⁸We do not mean that this module and its supporting theoretical research become obsolete. On the contrary, such research, which has become highly sophisticated nowadays, will contribute an immense amount to photogrammetry, cartography and other visual reconstruction problems. What we mean here is that if we can solve all the above-mentioned tasks, the general structure from motion module probably won't be needed for an autonomous "seeing" machine that is expected to perform navigational tasks.

object recognition in natural 3-D scenes, nor to deal with artificial scenes that contain large numbers of possible objects. Our work here falls into three broad categories:

- (a) Learning the appearance of an object.
- (b) Qualitative 3-D object recognition based on primitives.
- (c) Object recognition through recovery and matching.

4.1 Learning

How do we learn classes of shapes, for example, how a fish looks, given many examples of fishes? How do we learn to differentiate between different textures? How can we learn to detect significant changes in an area from aerial imagery? A system called ORACLE (ORganized Adaptive Constraint LEarning) has been constructed [Sullins, 1988a; 1988b; 1989a; 1989b] that can perform these tasks by learning the input-output behavior of a Boolean expression in disjunctive normal form.

Most methods of learning in distributed environments are based on gradient descent algorithms that involve changing the weights of the network in order to minimize the difference between the expected and actual input-output behaviors. The successes of such "motion in weight space" methods have been limited due to their inability to capture the implicit constraints of the behavior and properly distribute them among the units of the network. Our system is based on *motion in constraint space*. It relates the input-output behavior of a connectionist network to a Boolean expression in disjunctive normal form, where each hidden unit of the network learns to detect one of the conjunctive parts of the expression. The potential constraints at a processor are the states of an input configuration that correctly activates the outputs. These constraints are added and removed from the processors in such a way that the correctness of the behavior of the network is maximized. Unlike gradient descent methods, which may become trapped in local minima, or simulated annealing methods, which may need an infinite amount of time to reach a good state, this system determines a correct solution to many problems very quickly. Unlike most traditional "machine learning" algorithms, this system can learn concepts in parallel, is capable of continuously adapting to new information, and is highly resistant to feedback error.

Applications of this learning algorithm to tasks such as learning 2-D shapes from examples have demonstrated its potential applicability to practical problems. Recently, the algorithm has been generalized to learning under invariance. Also, it has been successfully used for texture discrimination.

The initial problem used to test ORACLE involved determining whether or not a given 6 by 6 pixel binary image (that is, a set of 36 inputs) contained a set of active input units in the shape of a square. There were 14 possible squares (9 of size 4, 4 of size 5, 1 of size 6),

and an input vector was given a 50% chance of being assigned one of these squares. Vectors containing squares were given "background noise" of 25%, that is, each non-square input was active with a probability of 25%. This means that there were on the order of 2^{24} (over 16 million) possible input vectors containing squares. The inputs of vectors not assigned squares were active with a probability of 50%, meaning that there were 2^{36} (over 60 billion) of these.

ORACLE learning curves for various levels of feedback error are shown in Figure 8. The X-axis represents the number of examples. The Y-axis represents the percentage of time that ORACLE gave the correct response to the question of whether or not the input contained a square. Note that this is not the percentage of time that the output matched the (possibly incorrect) feedback; we are interested in how well the system managed to ignore incorrect feedback, not in how well it duplicated it. Each curve is labeled with its level of feedback error, the percentage of time that the feedback was incorrect.

As Figure 8 shows, ORACLE learns to detect squares after seeing only a tiny fraction of the possible input vectors. In fact, its correctness is generally close to 100%, or at least much greater than the correctness of the feedback. This indicates that ORACLE succeeds in choosing the correct constraint motion in the long run despite occasional errors. While the learning time increases with the feedback error, the behavior is still learned quickly even for large amounts of error. The learning does not begin to deteriorate until the feedback error is greater than 30%.

The individual processors behave as predicted according to the theory. Each of them quickly acquires a set of correct constraints that distinguish it from the others and then goes through the slow process of removing the incorrect ones it picked up along the way. This is reflected in the learning curves, which

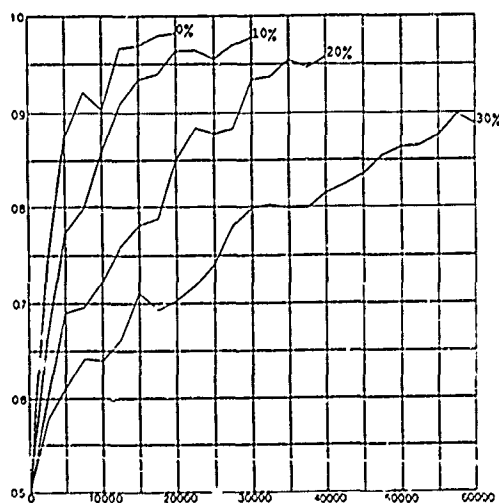


Figure 8.

quickly reach a good level of correctness and then slowly continue to improve to higher levels. They do not quite reach 100% correctness because incorrect constraints are still added from time to time.

The next experiment involved reducing the number of processors to 5, far below the minimum needed to cover all of the 14 conjunctive terms. The purpose of this was to force the network to learn *features* of squares. With a feedback error of 10%, the constraint sets in Figure 9 were created after 20,000 examples. As this figure shows, ORACLE discovered the ideas of *corners* and *parallel lines*. Both of these features allow a processor to accept more than one kind of square. Generalizing to these features did not increase the error significantly, as it is unlikely that the features would arise at random (2^{-7} for the corners, which is less than the 10% feedback error).

ORACLE was also given the more realistic problem of detecting fishtails. A set of 26 pictures of fish tails [Cousteau, 1953; 1963] were translated to the 6 by 6 binary format. These included many different species with greatly dissimilar tails, in order to insure that more than one type of detector would be needed. Background noise was added by activating inputs with probability 0.25, giving 2^{36} possible fish tails. ORACLE was either presented with one of these with probability .5 or was presented with random noise. Fish tails were given positive feedback and noise was given negative feedback, except for a feedback error of 10%. The network contained 10 processors.

The problem was made more interesting by also deactivating any input with probability 0.05. This means that there were no good conjunctive terms for the network to form, as there would always be fish tails that violated any set of constraints at the deactivated inputs. Because we allowed any of the inputs to be corrupted, ORACLE was forced to find the most basic prototypes of fish tails. These were far less well-defined than the features or conjunctive terms of the squares. Some of them are shown in Figure 10. The learning curve is shown in Figure 11.

As can be seen from Figure 10, ORACLE found widely varying features of fish tails. During each run, three or four processors acquired one of these features, but none of them accounted for a majority of the positive vectors that were accepted. This shows that the system was able to properly distribute the detection of different types of fish tails over different processors.

Figure 12 shows some textures that ORACLE was trained to discriminate and Figure 13 shows the

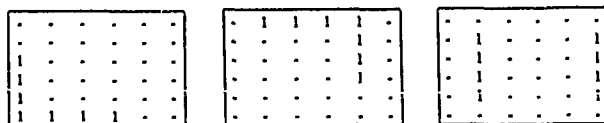


Figure 9.

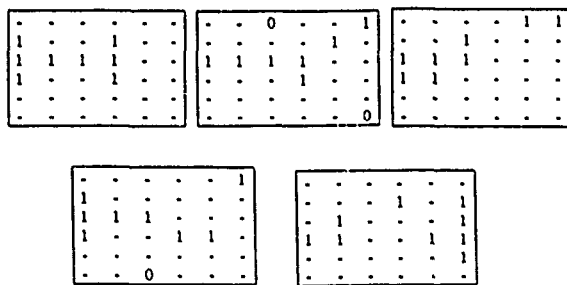


Figure 10.

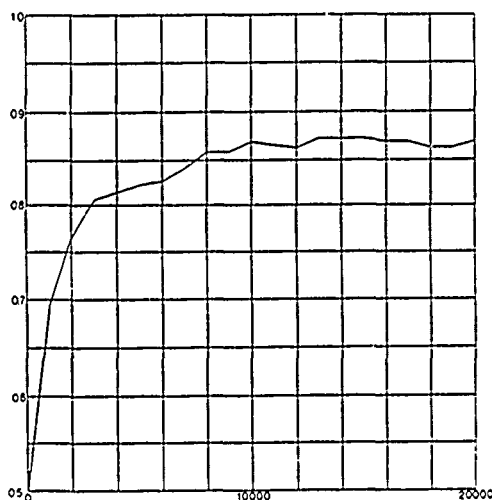


Figure 11.

corresponding learning curves.

4.2 Qualitative object recognition

Two important issues arise in the representation of objects for 3-D object recognition. The first issue is the choice between object-centered and viewer-centered representations. Object-centered representations model objects as constructions of 3-D primitives, such as planar faces or generalized cylinders. Viewer-centered representations model objects as a set of 2-D characteristic views, or aspects. The advantage of viewer-centered representation is that it reduces 3-D recognition to 2-D recognition; solving the inverse projection problem is unnecessary. However, with each model object having potentially many aspects, matching becomes less efficient than with object-centered models. The second issue concerns the amount of detail inherent in object models. Quantitative models facilitate simple, model-based verification procedures at the expense of model complexity. Qualitative models preclude top-down verification, but are invariant to minor changes in shape.

In [Dickinson et al., 1989] a modeling paradigm for 3-D object recognition integrating object-centered and viewer-centered models is proposed. Object models are object-centered constructions of 3-D volumetric

primitives, offering an efficient indexing mechanism for large object databases. The 3-D primitives, in turn, are mapped into a set of viewer-centered aspects. To minimize the size of the aspect set, the aspects are constrained to be invariant to minor changes in primitive shape, forcing the primitives to be qualitative in nature. Primitive reconstruction matches local 2-D image features to the set of viewer-centered aspects, whose size depends only on the size of the set of primitives, not on the number of object models or on object

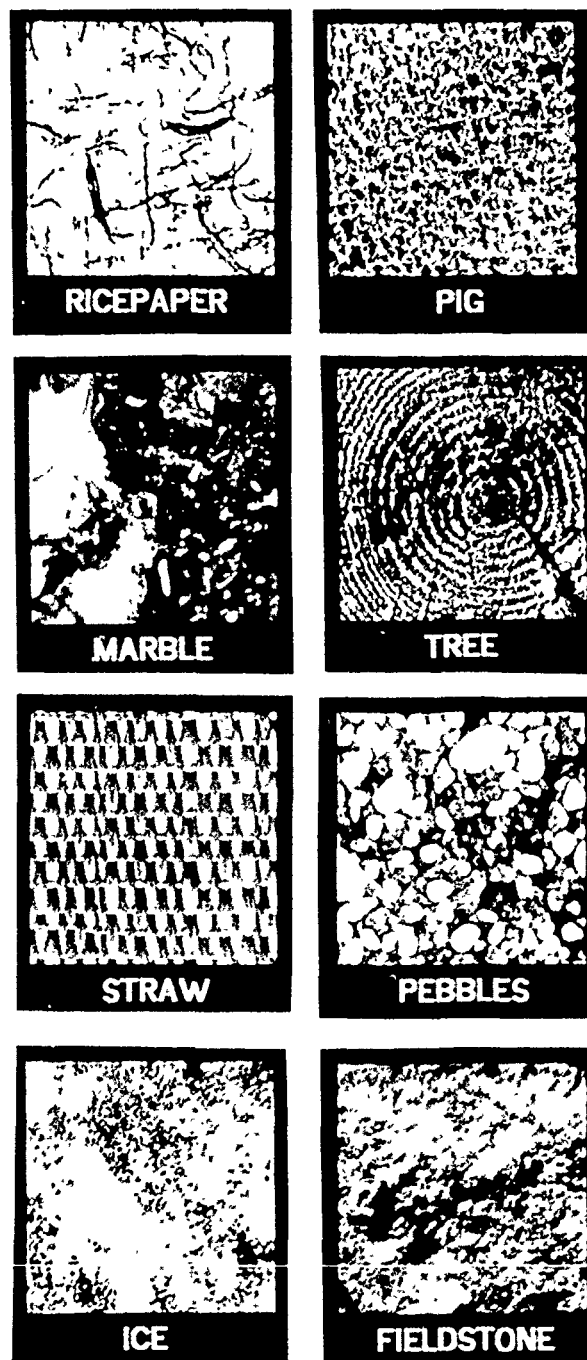


Figure 12.

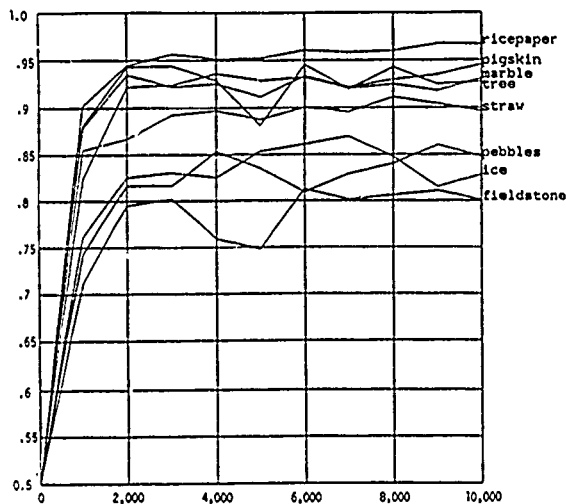


Figure 13.

model complexity. Object recognition then matches the primitives to object-centered models. To accommodate incomplete aspects arising from occluded model primitives, a hierarchical aspect representation based on aspect faces is introduced. The levels of the hierarchy are linked together by a set of conditional probabilities resulting from an extensive analysis of the aspects. This approach is general enough to allow for a wide choice of primitives. We are currently examining functional primitives, in accordance with the paradigm of purposive and qualitative vision.

4.3 Recovery and matching

We are continuing to study aspects of the matching process as well as specific recovery tasks—such as pose estimation, for example—related to object recognition. In [Margalit and Rosenfeld, 1989; Margalit and Knott, 1989; Kamgar-Parsi et al., 1989b] efficient algorithms are presented for matching polygonal arcs. The algorithm is based on an algorithm for run-length string matching presented in [Margalit and Rosenfeld, 1988].

In [DeMenthon and Davis, 1989] new exact and approximate solutions of the three-point perspective problems are presented. Model-based pose estimation techniques which match image and model triangles require large numbers of matching operations in real world applications. We have shown that by using approximations to perspective, lookup tables can be built for each of the triangles of the models. Weak perspective approximations have been previously applied to this problem; we have considered two other perspective approximations, paraperspective and orthoperspective. Analytical expressions are obtained which are as simple as those obtained using weak perspective, and which have much lower errors for off-center images than weak perspective. The errors are evaluated by comparison with exact solutions. The error estimates show the relative combinations of image and triangle characteristics which are likely to generate the largest errors. The corresponding cells of the lookup tables can be flagged, so that object pose calculations can

disregard image and model triangle pair combinations when their characteristics correspond to the flagged cells.

In [Friedland and Rosenfeld, 1989] a method was described for recognizing compact objects in an image by minimization of an energy function. The energy function is based on a polar coordinate object representation, define using any center from which the object's contour is visible. It incorporates both low-level and high-level information about the object: contour sharpness and smoothness at the low level, and contour shape at the high level.

5 Why Vision Is Hard: Vision Must Operate In Real Time

Suppose we ask a human to identify an object, and we measure the time T between the instant that the object is displayed and the instant at which it is identified. Let the average time for a neuronal firing, i.e. the time it takes a neuron to perform a computation and pass the result to its neighbors, be t . The quotient T/t is essentially the number of computational steps performed by the brain in order to identify the object. Amazingly, we find that this number is only a few hundred! Existing computer vision systems that perform non-trivial tasks require millions or billions of steps on a serial computer. Our work is addressing this issue by studying the role of parallel processing in computer vision.

[Chandran and Mount, 1989] developed optimal shared memory parallel algorithms for the Medial Axis Transform (MAT). In [Bestul, 1989] a general technique for defining SIMD Algorithms that operate on parallel pointer-based quadrees was developed. It is useful for creating parallel quadtree algorithms that run in time proportional to the height of the quadrees involved but that are independent of the number of objects (regions, points, segments, etc.) which the quadrees represent. The technique makes use of a dynamic relationship between the processors and the elements of the space and object domains being processed.

In [Sher and Rosenfeld, 1989] a pyramid programming environment on the Connection Machine is presented. The mapping between the Connection Machine and pyramid structures is based on a scheme called Shuffled 2-D Gray Codes. A pyramid Hough transform, based on computing the distances between line or edge segments and enforcing merge and select strategies among them, was implemented using this programming environment.

[Pehkonen, 1989] describes the implementation of a pose estimation algorithm on the Butterfly Parallel Processor (BPP) and Hathi 2 parallel computers. In [Davis and Narayanan, 1989] two approaches are described to the efficient processing of small images on hypercube-connected SIMD machines. The first approach, called fat images, is based on distributing the bits representing the gray level (or other feature) from each pixel across the processors of a sub-hypercube, using Gray coding techniques to obtain a

good mapping of the fat image into the hypercube. The second method, called replicated images, involves generating as many copies of the small image as will fit into the machine, and then distributing the computation of basic image processing operations across the copies.

A speed up analysis of parallel programs with an image analysis program as a case study was reported in [Seppänen, 1989]. The approach is based on the concept of critical paths of concurrent programs. Utilizing timing profiles of a program running on the desired target computer, a model can then speed up behavior of the program. The model can then be used for pointing out those factors that seem to affect the speed up behavior of the program. An image analysis algorithm was implemented on the Butterfly Parallel Processor (BPP) and used as a testbed for the method. The algorithm performs gray-level connected component analysis and feature extraction for area-segmented images. Then intra- and inter-area features are computed for the area segments, including adjacency graphs. Parallel computation is achieved by dividing the input image into equal sized blocks and assigning each block to a different processor of the BPP, according to the principles of data parallelism. An asynchronous slave process is attached to each image block, and an asynchronous master process controls the slaves via synchronization of barrier variables. Mutual exclusion among the slaves is implemented with locking primitives. Both of the main stages of the program contain three steps: first, image blocks are processed locally as long as possible; second, a description of block interfaces is created to be utilized in the third step, in which the partial results are merged.

In [Chen, 1989] a flexible parallel architecture for both discrete relaxation labeling (DRL) and probabilistic relaxation labeling (PRL) is developed. Through proper space-time arrangement of the computational steps involved, the relaxation labeling processes can be run on a systolic-array-like architecture in linear time for each iteration. Thus a high degree of computational parallelism is obtained. The arrays use one-dimensional, one-way communication lines between adjacent PEs and interface with the external environment through only a single I/O port. Because of the hardware simplicity and programmability features of the PEs, the architecture is well suited for VLSI implementation and is flexible enough to execute different relaxation algorithms. An illustrative example of running a region color labeling problem on the proposed architecture has been formulated and a general running procedure has also been developed.

6 Vision as a Part of a Larger System

The reason we need vision is to accomplish tasks. That is, vision is always a part of a larger system and we need to be able to integrate it with other cognitive abilities in a coherent and efficient manner. To be more concrete, let us consider the problem of visual navigation, i.e., visually mediated movement. This

requires us to consider vision in conjunction with planning. Most of the planning literature has concentrated on a complexity classification of various planning tasks. As most of the interesting problems are intractable, our research has focused on obtaining approximate efficient algorithms, i.e. algorithms that are not optimal but are proven efficient. For example, the problem of rearranging rectangular blocks enclosed in a rectangular room, or the *Warehouseman's Problem*, is known to be PSPACE-hard and hence is considered intractable. Following the suggestion in the seminal paper by Hopcroft, Schwartz and Sharir, we presented in [Sharma and Aloimonos, 1989] several constraints under which the general problem becomes tractable by giving polynomial algorithms which guarantee the rearrangement under different conditions. The concept of *temporary storage space* is introduced as an important part of the approach which can also be used in other hard motion coordination problems. The other constraints restrict the possible sizes and relative placements of the blocks.

Using the same methodology, the problem of finding a collision-free path connecting two points in the presence of obstacles, with constraints on the curvature of the path, is examined in [Basu and Aloimonos, 1989]. This problem of curvature-constrained motion planning arises when (for example) a vehicle with constraints on its steering mechanism needs to be maneuvered through obstacles. Though no lower bound on the difficulty of the problem in 2-D is known, the exact algorithms for the reachability question given so far are exponential. We have obtained a simple polynomial time algorithm for obtaining an approximation scheme for this problem. The approximation scheme can be used for obtaining the minimum curvature path or minimum length path satisfying a given curvature constraint. A probabilistic analysis of the scheme has also been given to analyze its usefulness.

We need to create a much closer coupling between vision and planning. Current experimental systems display a rigid distinction between their vision and planning modules, i.e., vision is used to extract information which is then used by the planner. We believe that vision and planning must be closely coupled, i.e. during planning, the vision module should be accessible at all times. Planning can also be seen as "taking actions that have some effect on the perceptual input". In this way, the planner can manipulate the input in a controlled manner. Using this viewpoint, [Basu, 1989] has investigated the problem of moving obstacles and on the basis of visual information has developed a computational theory that suggests several strategies that a robot can follow in order to plan a path (from a specified start to a specified end point) in the presence of moving obstacles whose motion is not known a priori. The input to this perceptual process is time varying imagery acquired by the robot and the output is a strategy that indicates how the robot should move in order to obtain a safe path, i.e. a strategy that maximizes the probability of safely reaching the goal using visually acquired knowledge at every instant.

Finally, based on this viewpoint about vision and planning, [Hervé et al., 1990] presents a robust technique for coordinating a hand/eye system without preliminary calibration of any of its components. Where the classical approach uses vision for calibrating the system as a step in the expensive inversion of the robot's kinematic map, we closely integrate the visual feedback in a qualitative control strategy to accomplish robot positioning tasks. The topology of the *perceptual kinematic map* between the joint coordinates of the robot and a set of image parameters is analyzed and exploited by a control strategy that provides the manipulator with an ability to successfully maneuver in its workspace.

7 Conclusions

We have presented a short summary of our image understanding research during the past year. Our discussion was organized around some fundamental reasons why vision is hard. These reasons are:

- *Ill-posedness of visual modules*, which we address through our work on discontinuous regularization, integration of modules and active vision.
- *Instability due to noise*, which we address through our work on provably optimal algorithms and the new paradigm of purposive and qualitative vision.
- *The difficulty of defining visual objects*, which we address through our work on learning.
- *The fact that vision must be real-time*, which we address through our research on parallel algorithms and architectures.

We also emphasize the fact that vision must be part of a larger system. In particular, we propose coupling vision and planning in a strong way, so that they operate together at all times. Our work on navigation demonstrates that this is both feasible and very efficient.

References

- [Aloimonos, 1988] J. Aloimonos, "Shape from texture", *Biological Cybernetics* **58**, 345-360, 1988.
- [Aloimonos, 1989] J. Aloimonos, "Unifying shading and texture through an active observer", *Proc. Royal Soc. London B* **238**, 25-27, 1989.
- [Aloimonos and Hervé, in press] J. Aloimonos and J.-Y. Hervé, "Correspondenceless detection of depth and motion for a planar surface", *IEEE Transactions on PAMI*, in press.
- [Aloimonos and Huang, 1990] J. Aloimonos and L. Huang, "Motion-boundary illusions and their regularization", Technical Report CAR-TR-495, Center for Automation Research, University of Maryland, College Park, 1990.
- [Aloimonos and Shulman, 1987] J. Aloimonos and D. Shulman, "Learning early vision computations", *J. Opt. Soc. Am., A* **6**, 908-919, 1987.
- [Aloimonos and Shulman, 1989] J. Aloimonos and D. Shulman, *Integration of Visual Modules: An Extension of the Marr Paradigm*, Academic Press, Boston, 1989.
- [Ballard, 1989] D.H. Ballard, "Animate vision", *Proc. IJCAI*, 1989.
- [Bandopadhyay, 1986] A. Bandopadhyay, Ph.D. Thesis, Dept. of Computer Science, University of Rochester, 1986.
- [Basu, 1989] A. Basu, "A framework for motion planning in the presence of moving obstacles", Technical Report CAR-TR-481, Center for Automation Research, University of Maryland, College Park, 1989.
- [Basu and Aloimonos, 1987] A. Basu and J. Aloimonos, "A robust algorithm for determining the translation of a rigidly moving surface without correspondence for robotics application", *Proc. IJCAI*, 1987.
- [Basu and Aloimonos, 1989] A. Basu and J. Aloimonos, "Approximate constrained motion planning", Technical Report CAR-TR-435, Center for Automation Research, University of Maryland, College Park, 1989.
- [Bestul, 1989] T. Bestul, "A general technique for creating SIMD algorithms on parallel pointer-based quadrees", Technical Report CAR-TR-420, Center for Automation Research, University of Maryland, College Park, 1989.
- [Blake and Zisserman, 1987] A. Blake and A. Zisserman, *Visual Reconstruction*, M.I.T. Press, Cambridge, MA 1987.
- [Brady, 1982] M. Brady, "Computational approaches to image understanding", *ACM Computing Surveys* **14**, 1982.
- [Brosh et al., 1989] M. Brosh, B. Kamgar-Parsi and B. Kamgar-Parsi, "Reliability analysis of the closed-form solution to the image flow equations for 3D structure and motion (planar patch)", Technical Report CAR-TR-431, Center for Automation Research, University of Maryland, College Park, 1989.
- [Chandran and Mount, 1989] S. Chandran and D. Mount, "Optimal shared memory parallel algorithms and the medial axis transform", Technical Report CAR-TR-411, Center for Automation Research, University of Maryland, College Park, 1989.
- [Chen, 1989] Z. Chen, "A flexible parallel architecture for relaxation labeling algorithms", Technical Report CAR-TR-474, Center for Automation Research, University of Maryland, College Park, 1989.
- [Cousteau, 1953] J.Y. Cousteau, *The Silent World*, Harper and Row, New York, 1953.
- [Cousteau, 1963] J.Y. Cousteau, *The Living Sea*, Harper and Row, New York, 1963.
- [Davis and Narayanan, 1989] L.S. Davis and P.J. Narayanan, "Efficient multiresolution image processing on hypercube connected SIMD machines",

- Technical Report CAR-TR-430, Center for Automation Research, University of Maryland, College Park, 1989.
- [DeMenthon and Davis, 1989] D. DeMenthon and L.S. Davis, "New exact and approximate solutions of the three-point perspective problem", Technical Report CAR-TR-471, Center for Automation Research, University of Maryland, College Park, 1989.
- [Dickinson et al., 1989] S.J. Dickinson, A.P. Pentland and A. Rosenfeld, "A representation for qualitative 3-D object recognition integrating object-centered and viewer-centered models", Technical Report CAR-TR-453, Center for Automation Research, University of Maryland, College Park, 1989.
- [Duncan and Li, 1989] J. Duncan and L. Li, "The computation of general 3-D motion without correspondence from binocular optical flows", Technical Report CAR-TR-452, Center for Automation Research, University of Maryland, College Park, 1989.
- [Feldman, 1985] J.A. Feldman, "Four frames suffice: A provisional model of vision and space", *Behavioral and Brain Sciences* 8, 265-313, 1985.
- [Friedman and Rosenfeld, 1989] N. Friedland and A. Rosenfeld, "Compact object recognition using function based optimization", Technical Report CAR-TR-478, Center for Automation Research, University of Maryland, College Park, 1989.
- [Geman and Geman, 1984] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distribution and Bayesian Restoration of Images," *IEEE Transactions on PAMI* 6, 1984.
- [Gibson, 1950] J.J. Gibson, *The Perception of the Visual World*, Houghton-Mifflin, Boston, 1950.
- [Hadamard, 1923] J. Hadamard, *Lectures on the Cauchy Problem in Linear Partial Differential Equations*, Yale University Press, New Haven, CT, 1923.
- [Hervé and Aloimonos, 1990] J.-Y. Hervé and J. Aloimonos, "Shading into texture and texture into shading", *Proc. 1st ECCV*, 1990.
- [Hervé et al., 1990] J.-Y. Hervé, R. Sharma and P. Cucka, "Qualitative visual control of a robot manipulator", Technical Report, Center for Automation Research, University of Maryland, College Park, to appear, 1990.
- [Horn, 1986] B.K.P. Horn, *Robot Vision*, M.I.T. Press, Cambridge, MA, 1986.
- [Horn and Schunck, 1981] B. Horn and B. Schunck, "Determining Optical Flow," *Artificial Intelligence* 17, 185-204, 1981.
- [Huber, 1981] P. Huber. *Robust Statistics*. Wiley. New York, 1981.
- [Jasinschi, 1989a] R.S. Jasinschi, "Intrinsic constraints in space-time filtering: A new approach to representing uncertainty in low-level vision", Technical Report CAR-TR-425, Center for Automation Research, University of Maryland, College Park, 1989.
- [Jasinschi, 1989b] R.S. Jasinschi, "Energy filters, motion uncertainty, and motion sensitive cells in the visual cortex: A mathematical analysis", Technical Report CAR-TR-487, Center for Automation Research, University of Maryland, College Park, 1989.
- [Julesz, 1971] B. Julesz, *Foundations of Cyclopean Perception*, Univ. of Chicago Press, Chicago, 1971.
- [Kamgar-Parsi et al., 1989a] B. Kamgar-Parsi, B. Kamgar-Parsi and W.A. Sander, "Quantization error in spatial sampling: comparison between square and hexagonal pixels", Technical Report CAR-TR-415, Center for Automation Research, University of Maryland, College Park, 1989.
- [Kamgar-Parsi et al., 1989b] B. Kamgar-Parsi, A. Margalit and A. Rosenfeld, "Matching general polygonal arcs", Technical Report CAR-TR-442, Center for Automation Research, University of Maryland, College Park, 1989.
- [Land and McCann, 1971] E.H. Land and J.J. McCann "Lightness and retinex theory", *J. Opt. Soc. Am.* 61, 1-11, 1971.
- [Margalit and Rosenfeld, 1988] A. Margalit and A. Rosenfeld, "Reducing the expected computational cost of template matching using run length representation", Technical Report CAR-TR-406, Center for Automation Research, University of Maryland, College Park, 1988.
- [Margalit and Rosenfeld, 1989] A. Margalit and A. Rosenfeld, "Matching polygonal arcs", Technical Report CAR-TR-418, Center for Automation Research, University of Maryland, College Park, 1989.
- [Margalit and Knott, 1989] A. Margalit and G. Knott, "An algorithm for computing the union, intersection or difference of two sets of polygons", Technical Report CAR-TR-419, Center for Automation Research, University of Maryland, College Park, 1989.
- [Marr, 1982] D. Marr, *Vision*, W.H. Freeman, San Francisco, 1982.
- [Marroquin, 1985] J. Marroquin, "Probabilistic solution of inverse problems," Ph.D. Thesis, M.I.T., 1985.
- [Meer and Weiss, 1989] P. Meer and I. Weiss, "Smoothed differentiation filters for images", Technical Report CAR-TR-424, Center for Automation Research, University of Maryland, College Park, 1989.
- [Meer et al., 1989] P. Meer, D. Mintz and A. Rosenfeld, "Least median of squares based robust analysis of image structure", Technical Report CAR-TR-490, Center for Automation Research, University of Maryland, College Park, 1989.
- [Mumford and Shah, 1985] D. Mumford and J. Shah, "Boundary detection by minimizing functionals," *Proc. IEEE CVPR*, 22-25, 1985.
- [Nelson and Aloimonos, 1988] R.C. Nelson and J. Aloimonos, "Towards qualitative vision: Using flow field divergence for obstacle avoidance in visual

- navigation", *Proc. 2nd IEEE ICCV*, 1988.
- [Pehkonen, 1989] K. Pehkonen, "The implementation of Linnaïmaa and Harwood's pose determination algorithm on shared and distributed memory parallel computers", Technical Report CAR-TR-423, Center for Automation Research, University of Maryland, College Park, 1989.
- [Poggio et al, 1984] T. Poggio, H. Voorhees, and A. Yuille, "Regularizing edge detection," AI Memo 776, M.I.T. Artificial Intelligence Laboratory, 1984.
- [Poggio et al., 1985] T. Poggio, V. Torre and C. Koch, "Computational vision and regularization theory", *Nature* **317**, 214-319, 1985.
- [Ramachandran, 1989] V.S. Ramachandran, invited talk at IEEE Workshop on Visual Motion, 1989.
- [Regan] D. Regan, personal communication.
- [Searle, 1984] J. Searle, *Intentionality*, Cambridge University Press, Cambridge, UK, 1984.
- [Seppänen, 1989] T. Seppänen, "Speed-up analysis of parallel programs with an image analysis program as a case study", Technical Report CAR-TR-459, Center for Automation Research, University of Maryland, College Park, 1989.
- [Sharma and Aloimonos, 1989] R. Sharma and J. Aloimonos, "Resolving the intractability of the Warehouseman's Problem using temporary storage space and other constraints", Technical Report CAR-TR-451, Center for Automation Research, University of Maryland, College Park, 1989.
- [Sher and Rosenfeld, 1989] C.A. Sher and A. Rosenfeld, "A pyramid Hough transform on the Connection Machine", Technical Report CAR-TR-421, University of Maryland, College Park, 1989.
- [Shulman, to appear] D. Shulman, Ph.D. Thesis, University of Maryland, College Park.
- [Shulman and Aloimonos, 1988a] D. Shulman and J. Aloimonos, "(Non)rigid motion interpretation", *Proc. Royal Soc. London B*, **233**, 217-234, 1988.
- [Shulman and Aloimonos, 1988b] D. Shulman and J. Aloimonos, "Boundary preserving regularization: Theory, Part I" Technical Report CAR-TR-340, Center for Automation Research, University of Maryland, College Park, 1988.
- [Shulman and Hervé, 1990] D. Shulman and J.-Y. Herve, "On the regularization of discontinuous flow fields", Technical Report CAR-TR-498, Center for Automation Research, University of Maryland, College Park, 1990.
- [Spetsakis and Aloimonos, 1988a] M.E. Spetsakis and J. Aloimonos, "Optimal computing of structure from motion using point correspondences in two frames", *Proc. 2nd IEEE ICCV*, 1988.
- [Spetsakis and Aloimonos, 1988b] M.E. Spetsakis and J. Aloimonos, "Optimal computing of structure from motion using point correspondences in two frames", Technical Report CAR-TR-389, Center for Automation Research, University of Maryland, College Park, 1988.
- [Spetsakis and Aloimonos, 1988c] M.E. Spetsakis and J. Aloimonos, "A multi-frame approach to visual motion perception", Technical Report CAR-TR-407, Center for Automation Research, University of Maryland, College Park, 1988.
- [Spetsakis and Aloimonos, in press] M.E. Spetsakis and J. Aloimonos, "Closed form solution to the structure from motion problem using line correspondences", *IJCV*, in press.
- [Spetsakis and Aloimonos, 1989] M.E. Spetsakis and J. Aloimonos, "Unification theory of structure from motion", Technical Report CAR-TR-482, Center for Automation Research, University of Maryland, College Park, 1989.
- [Sullins, 1988a] J.R. Sullins, "Boolean learning in neural networks", Technical Report CAR-TR-359, Center for Automation Research, University of Maryland, College Park, 1988.
- [Sullins, 1988b] J.R. Sullins, "Distributed learning: motion in constraint space", Technical Report CAR-TR-412, Center for Automation Research, University of Maryland, College Park, 1988.
- [Sullins, 1989a] J.R. Sullins, "Distributed learning of texture classification", Technical Report CAR-TR-444, Center for Automation Research, University of Maryland, College Park, 1989.
- [Sullins, 1989b] J.R. Sullins, "Distributed learning under invariance", Technical Report CAR-TR-479, Center for Automation Research, University of Maryland, College Park, 1989b.
- [Ullman, 1979] S. Ullman, *The Interpretation of Visual Motion*, M.I.T. Press, Cambridge, MA, 1979.
- [Weiss, 1989] I. Weiss, "Image smoothing and differentiation with minimal curvature filters", Technical Report CAR-TR-470, Center for Automation Research, University of Maryland, College Park, 1989.
- [Yuille and Grzywacz, 1988] A.L. Yuille and N. Grzywacz, "A motion coherence theory", *Proc. 2nd ICCV*, 1988.

USC IMAGE UNDERSTANDING RESEARCH: 1989-1990

R. Nevatia, K. Price and G. Medioni*
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, California 90089-0273

Abstract

This paper summarizes the USC Image Understanding research projects and provides references to more detailed sources of information. Our work has focussed on the topics of 3-D vision (including range data processing, stereo, shape from contour and object recognition), aerial image analysis, motion analysis (including spatio-temporal analysis, 3-D motion estimation, detection of moving objects and an integrated motion system), and parallel processing (including mapping algorithms onto specific or flexible architectures, and processor-time tradeoffs).

1 INTRODUCTION

This paper summarizes our research projects during the last year. Some of this work is described in more detail in other papers in these proceedings [Stein and Medioni, 1990b; Menet *et al.*, 1990; Kim and Price, 1990; Ulupinar and Nevatia, 1990b; Reinhart and Nevatia, 1990; Frazier and Nevatia, 1990]; this work is covered only briefly in this summary. We also provide references to details for work not described elsewhere in these proceedings.

Our research activity has focussed on the following major topics:

- 3-D Vision
- Aerial Image Analysis
- Motion Analysis, and
- Parallel Processing

In all of these areas, we have had broad research programs that have been carried out for an extended period of time. Thus, it is not possible for us to give a complete summary of our work here. Rather, we describe our new results and attempt to give some context of the long term work in which these results fit.

*This research was supported by the Defense Advanced Research Projects Agency under contracts F33615-87-C-1436 and F49620-89-C-0126, monitored by the Wright-Patterson Air Force Base and the Air Force Office of Scientific Research, respectively.

2 3-D VISION

Our goal here is to develop techniques for description and recognition of complex 3-D objects in complex scenes. We focus on the analysis of objects using shape (as opposed to texture or other cues) and have made significant progress in the past year. In particular, we have concentrated on the following:

- *Range image analysis*
We have made progress in the automatic acquisition of models from multiple views, using either symbolic or iconic representations. These models are useable for a variety of applications, including for object recognition as in a system described in our previous work [Fan *et al.*, 1989].
- *Stereo*
 - We have completed a system which combines area-based and feature-based processing to generate dense disparity maps.
 - We have excellent results performing the matching using very high level primitives resulting from perceptual organization
 - In the special case of urban scenes, we have used “snakes” to accurately delineate the contours of building tops.
- *Shape from contour*
We have developed a theory for inferring 3-D shape of objects from their contours. The technique relies on observations of certain types of symmetries in the contours and the mathematical constraints that derive from them. Our technique uses relatively few assumptions and heuristics and is largely based on geometrical properties of contours. We have shown that it is applicable to the analysis of zero-Gaussian curvatures surface, straight homogeneous generalized cylinders and “snakes” and are working on extending it to yet more complex objects. Good results are obtained, however, currently we assume that contours and symmetries are given to our system. In separate projects, we are investigating the computation of such symmetries.
- *Symmetry Detection and Perceptual Grouping*
Grouping of contours detected in an image is crucial in proper segmentation and description of objects in

a scene. In our previous work, we found that symmetries play a key role in computing such perceptual groupings [Rao, 1988; Mohan, 1989]. Symmetries are also central to our technique for inferring 3-D shape from contours. In recent work, we have been investigating efficient ways of computing these symmetries [Saint-Marc and Medioni, 1990]: once edge contours are represented by approximating B-splines and the corners detected [Saint-Marc *et al.*, 1989], the computation of symmetries is of complexity $O(n^2)$, where n is the number of spline segments as opposed to the number of points.

- **Matching**

We have defined a methodology based on efficient coding and hash tables to recognize objects in a cluttered environment, even when the number of models is large. We can successfully recognize flat objects under affine transform, and 3-D objects given 3-D data (such as range images), with no restrictive assumptions on the shape of these objects.

2.1 RANGE IMAGE ANALYSIS

Range imagery differs from intensity imagery in that the input directly relates to the geometric *shape* of the objects in the scene. Our previous work has allowed us to compute symbolic descriptions of range images, and to perform matching with multi-view models. Recently, we have obtained integrated representations of models from multiple views, which is more natural since such models can be observed offline from many positions. The model building procedure is performed either by merging at the data level prior to segmentation, or by merging the segmented views, as explained below.

2.1.1 Data level merging

One of the difficulties of integrating multiple views is in finding an accurate transformation between data obtained from different views. Previous research has suggested to determine the relative motion between views by using marks and regular patterns in the scene by taking intensity images at the same time and matching those features [Vemuri and Aggarwal, 1986], or by matching surface features directly [Ferrie and Levine, 1987]. These techniques rely solely on the accuracy of feature detection and provide no feedback from the data themselves as to how well the different views have been registered under the estimated transformation.

Our approach is to use range data directly and try to register successive views of the object with overlapping areas to compute transformations for the relative motion between views. To reduce the possible large search space and ensure that the algorithm converges, we assume that the approximate transformation between the data from two views is known, which is reasonable when the range data are acquired in a controlled environment.

To register two overlapping views of range image of the object, we first choose a set of surface points, called control points, from one of the range image, and then apply a minimization process to find the rigid transformation which minimizes a distance measure from those control points to the surface represented by the other

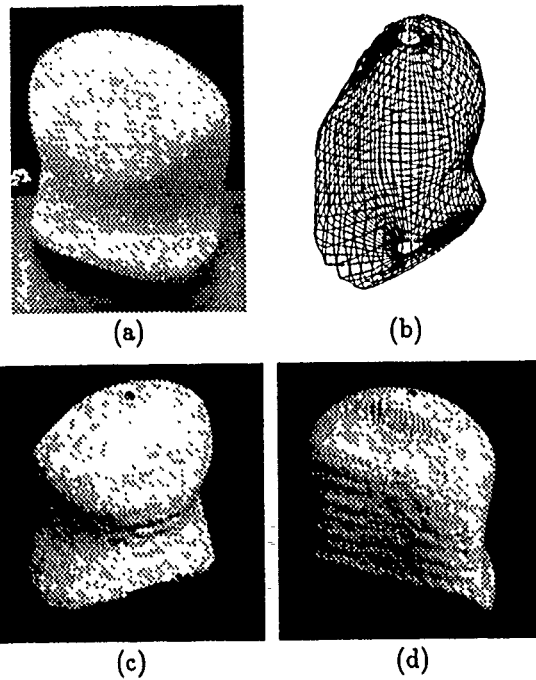


Figure 1: Object Modeling:
(a) The original wood block, (b) wireframe of the reconstructed model and (c) & (d) two rendered images of the model

range images. This minimization process is done by using least-square method iteratively. The control points and the distance measure have been chosen so that this process can converge very fast.

To merge multiple views, we use a simple cylindrical/spherical representation for simple compact objects. Successive range image of views of the object are merged after being mapped to a object-centered coordinate frame by using the relative transformations found by the registration process. To avoid the introduction of a cumulative error term in the integration process, we also use a global registration strategy, i.e., we always register the next view with the integrated result so far, so that each view of range data can be registered more globally with the remaining views.

An example is illustrated in Figure 1: the wood block a) has been viewed from 8 side positions 45° apart, from the top and the bottom. The reconstructed views of the object are shown as shaded images in b) and c).

2.1.2 Symbolic level merging

The alternative approach consists of generating a symbolic description, such as an attributed graph, for each view, and then merge the different descriptions at this high level. Each view is represented by a graph whose *nodes* are the individual surface patches and the *links* are the relationships between adjacent patches. The matching between views is achieved either through a tree search procedure [Fan *et al.*, 1989], or by a 2-level constraint satisfaction network [Parvin and Medioni, 1989]. One of the difficulties to be overcome by this process is the inference of surface patches from bounding contours, since these are not necessarily continuous and generally

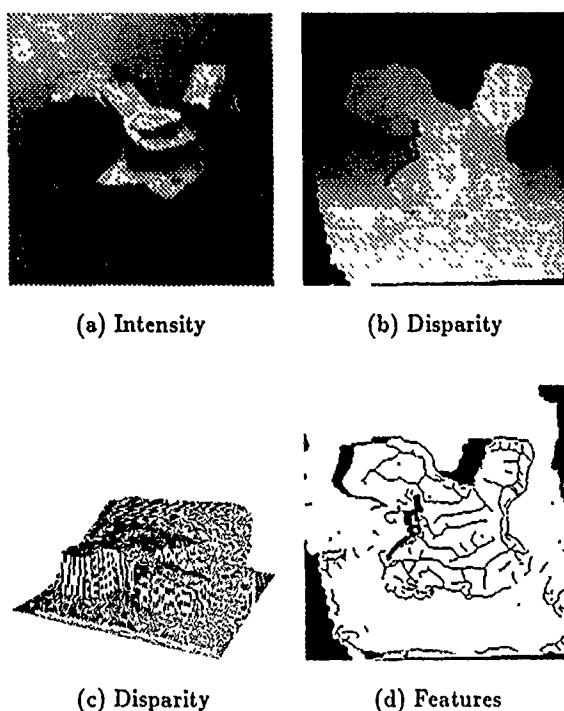


Figure 2: Renault Part.

inaccurate at junctions. We have obtained good results by modeling this process as a dynamic network subject to *weak smoothness* constraints. The initial state of the network consists of the curves produced by low level operators, but these decay over time unless excited. Possible completions provide this excitation, competing with each other and strengthening existing curves [Parvin and Medioni, 1990].

2.2 STEREO

We are using different approaches to solving the stereo correspondence problem, from using a combination of area-based and feature-based processing, to working with complex primitives resulting from a perceptual grouping stage. We also are using active contours to obtain accurate boundaries of roof tops in aerial views of urban areas.

2.2.1 Feature and area-based processing

We have considerably improved the system described last year [Cochran and Medioni, 1989a], in which we integrate area-based and feature-based processing, taking advantage of the unique attributes provided by each one separately. The area-based processing generates a dense disparity map, and the feature-based processing accurately locates discontinuities. The first improvement, described in [Cochran and Medioni, 1989b], is the extraction of depth and, in many cases, orientation discontinuities from the image.

Figure 2 shows the results obtained for the "Renault Part" stereo pair. Figure 2 (a) and (b) show one of the

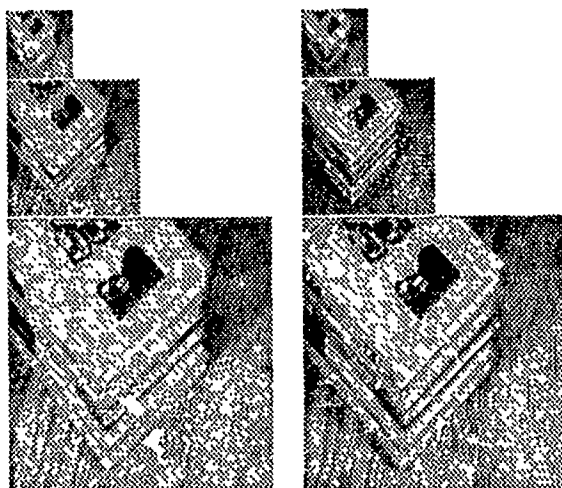


Figure 3: Books — Intensity Pyramid.

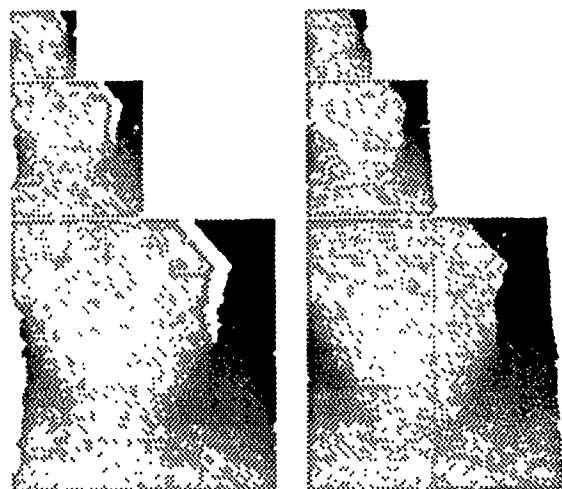


Figure 4: Books — Disparity Pyramid.

stereo intensity images and the respective disparity result; (c) shows a 3-D plot of the disparity, from which the surface features (d) were extracted. The surface features located on the disparity surface are the depth discontinuities, the occluded regions, and the concave and convex folds.

The second improvement is the use of a multi-level pyramid, first processing a reduced (coarse) version of the image pair, and then propagating the results to another level for higher-resolution (finer) processing, as shown in figures 3 and 4. This introduces a more global context and allows the correction of local errors in matching, such as those due to photometric and geometric distortions. Figure 5 shows a 3-D plot of the disparity and figure 6 shows the extracted surface features.

We have applied this Stereo Vision System to a wide variety of scenes and obtained results which compare very favorably with state-of-the-art methods [Olsen, 1990; Hoff and Ahuja, 1989; Drumheller and Poggio,

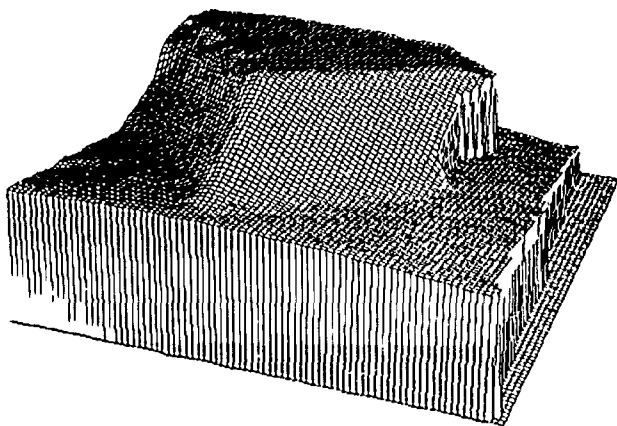


Figure 5: Books — 3-D Plot of Disparity.

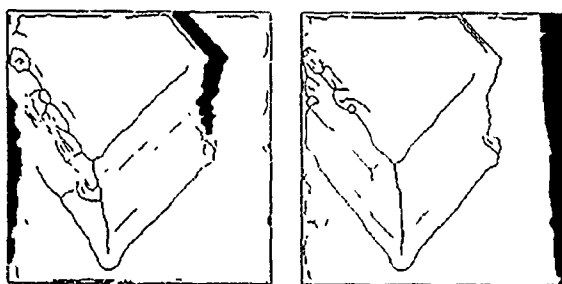


Figure 6: Books — Surface Features.

1986].

2.2.2 Stereo of aerial urban scenes

Current stereo algorithms, whether area-based or feature-based, tend to fail around depth discontinuities, since these are the locations where smoothness assumptions do not hold. This phenomenon is most easily observable in aerial views of urban scenes, and the roofs of buildings can therefore be detected, but not accurately delineated. Fua [Fua and Leclerc, 1988] and Mohan [Mohan and Nevatia, 1989b] propose to solve the problem by restricting the possible shapes in the form of a generic model.

Here instead, we propose to use the initial estimate provided by a traditional stereo system (as described in the last section). and to refine it by enforcing a local smoothness constraint. This is accomplished by an active contour model, whose details are given in these proceedings [Menet *et al.*, 1990]. The estimate is shown in figures 7-10.

We have obtained excellent results, even when the boundaries contain corners, as illustrated on figure 11 below.

2.2.3 Stereo matching using high level features

We are also investigating an alternative approach to stereo that uses high level features for correspondence. Lower level feature matching may have difficulties with global correspondence, particularly when repetitive

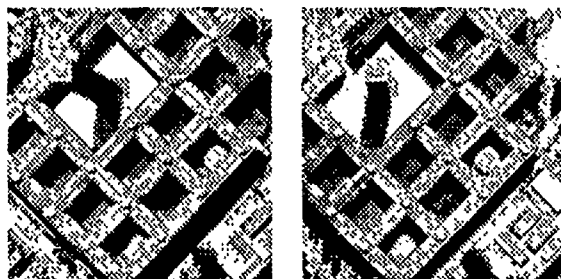


Figure 7: Jussieu — Intensity.

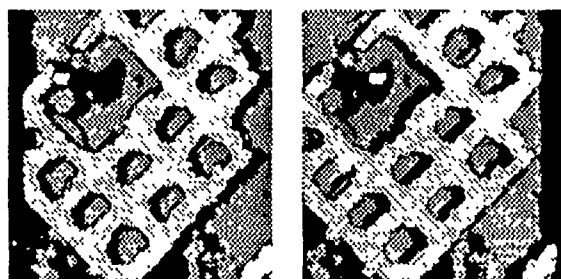


Figure 8: Jussieu — Disparity.

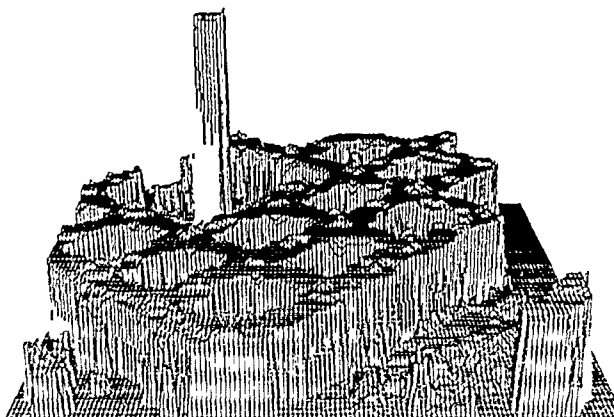


Figure 9: Jussieu — 3-D Plot of Disparity.

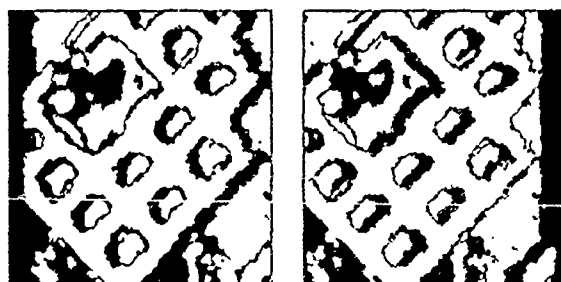


Figure 10: Jussieu — Surface Features.

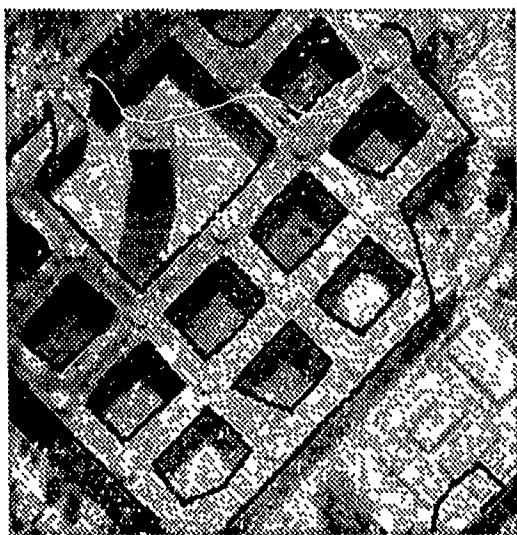


Figure 11: Example of delineation of buildings roofs with deformable contour models

structures are present, requires presence of rather dense texture and highly accurate knowledge of epipolar geometry. High level feature matching can potentially overcome these obstacles. Further high level features are fewer in number and hence should be faster to match, at least in principle. However, this approach has the deficiency that high level features need to be computed from monocular images; a process that is well known to be difficult and error prone in itself. We have developed sophisticated perceptual grouping methods to overcome this difficulty [Mohan, 1989].

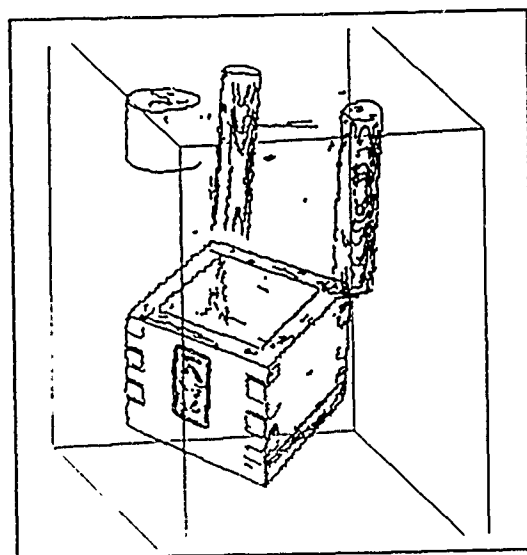
Our first experience with high level stereo was in the context of analyzing buildings in aerial scenes. In such scenes, texture (on the roofs) is very sparse and disparity changes discontinuously at the boundaries. We found that using high level features (rectangles) was very effective for stereo processing of such scenes [Mohan and Nevatia, 1989b]. We next investigated generalization of this approach to scenes where the object shape is not so constrained [Mohan and Nevatia, 1989b]. In this work, we found that ribbons (defined by two symmetrical curves with closures at the two ends) are an effective method for organizing the curves in an image into higher level features and that these ribbons could be used for stereo matching. This work, however, concentrated on the grouping problem and not on development of a competent stereo system.

In our recent work, we have been building on our perceptual grouping system to develop a stereo system. Features such as edges, curves, symmetries, and ribbons which represent geometric structures of objects in the scene are extracted from each image using perceptual grouping. The grouping algorithms are similar to those described in [Mohan and Nevatia, 1989b] but several enhancements have been incorporated. Hierarchy of features from the left and right images are then matched using a relaxation network. Our method has shown ac-



(a)

(b)



(c)

Figure 12: Results of a scene with multiple occlusions (a) left image (b) right image (c) Disparity output

curate results for images of multiple occlusions and wide angle disparities. This is illustrated on figure 12.

2.3 3-D SHAPE FROM CONTOURS

Humans are able to readily perceive 3-D shape from a monocular image. Many cues are used in this process such as shading, shadows and texture. However, we believe that the most significant cue is the shape of the 2-D contours. The process of inferring 3-D shape from contours, however, has proven to be a very difficult one. We believe that we have made a major advance in this area and developed a theory that extends the range of shapes that can be analyzed significantly. Our theory relies on observations of certain symmetries in the scene and we conjecture that only shapes having some symmetries are perceived in 3-D by humans as well.

We define two types of symmetries that we call *parallel* and *mirror* symmetries (the precise definitions are given in another paper in these proceedings [Ulpinar and Nevatia, 1990b]). Given the observations of these symmetries in some specific combination, we can infer

some qualitative properties of surfaces and objects in the scene, such as whether they are planar, have a zero-Gaussian curvature surface, or are some specific classes of generalized cylinders.

Further, the contours and the symmetries allow us to formulate some constraints on the quantitative shape for the surfaces being viewed. The constraints that derive purely from the geometry of the surface are, however, not sufficient to compute the precise shape of the surface and leave some degrees of freedom unconstrained. These degrees of freedom can also be fixed by using some simple perceptual properties.

Our technique is rather mathematical and hence difficult to summarize without introducing a good deal of notation. Hence, here we only give references to the more detailed work and show some examples. The basics of our method, and its applications to analysis of zero-Gaussian curvature surfaces are given in [Ulpinar and Nevatia, 1990a]. Figure 13 shows some examples from this work. The first column of this figure shows the input contours to the program, the middle column shows the computed surface orientations as a "needle diagram" and the last column shows the surface orientations by painting the surface with intensities that would result from a Lambertian surface illuminated by a point source. Extensions of our method to straight homogeneous generalized cylinders (SHGCs) and snakes (generalized cylinders of constant cross-section) and some results are given in [Ulpinar and Nevatia, 1990b].

We hope that these examples indicate the power and range of our approach. We are in the process of further developing the theory to apply to yet more complex objects. It should be noted that this technique assumes that the appropriate contours and symmetries are given; this is far from a trivial task. However, we are making progress on detection of the appropriate symmetries in other projects in our group [Mohan and Nevatia, 1989a; Saint-Marc and Medioni, 1990].

2.4 SYMMETRY DETECTION

Once edges are extracted, the resulting contours need to be represented for further reasoning. Iconic representations do not make the necessary information explicit: by definition edges only capture very local properties of an image, and the inference of higher structures, such as object boundaries, requires *grouping* operations. We believe that such operations rely on basic and simple properties and various forms of symmetry [Mohan, 1989]. The representation must therefore make explicit differential properties of contours, such as tangent and curvature. Furthermore, because of the variability inherent in the imaging process, the representation should be tolerant to noise, partial occlusion, and perspective, naturally suggesting segmented, local descriptors [Rao *et al.*, 1987].

If the world was composed of polyhedral objects alone, we would know to expect only straight line segments in images, and polygonal approximations would be appropriate. In many cases, such an approximation is indeed sufficient, as demonstrated by several applications such as stereo [Medioni and Nevatia, 1985], aerial im-

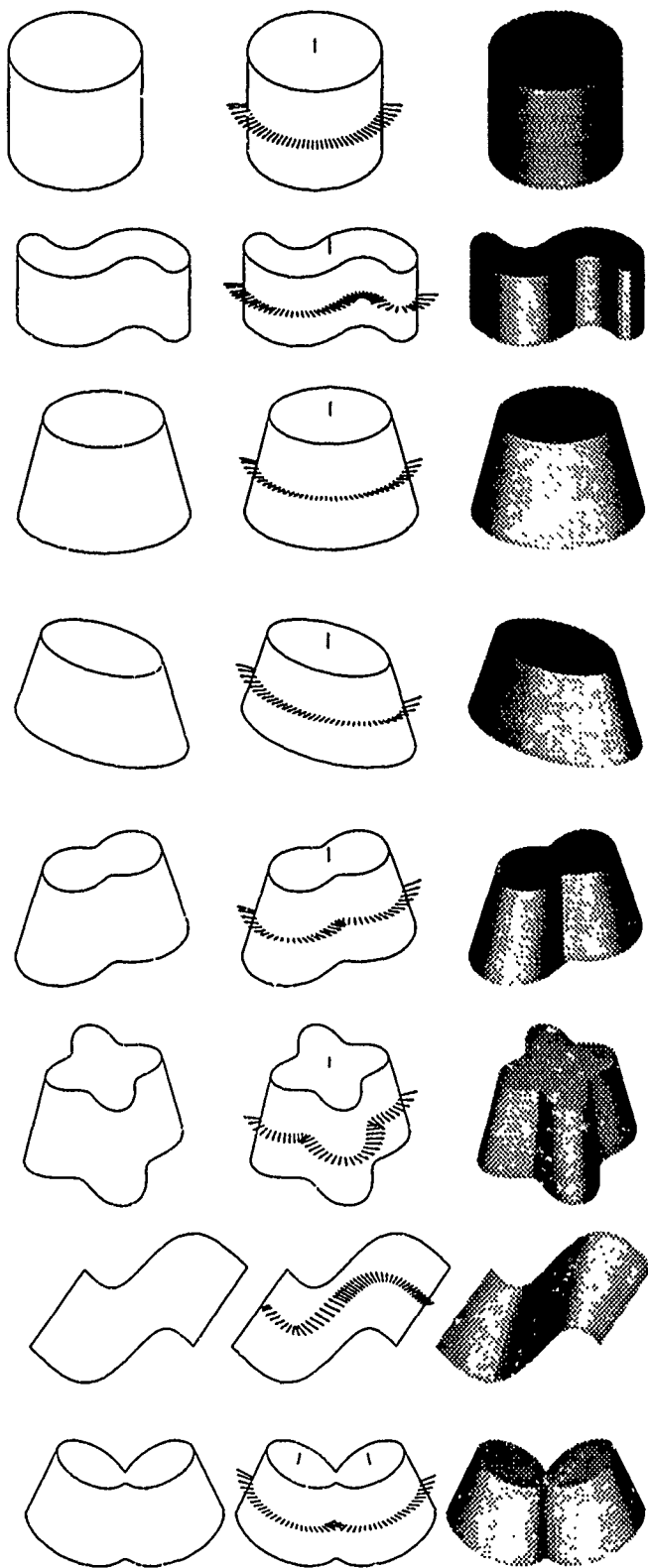


Figure 13: Sample contours, the needle images computed and their images after shading the object with the computed orientation at every point on the surface. The last object has a non planar cross section and thus it is segmented into two planar cross section objects before processing.

age understanding [Huertas and Nevatia, 1988a] or object recognition [Mundy *et al.*, 1988; Stein and Medioni, 1990a], but is unable to capture curvature information, since it is a first order approximation. Also, if a contour is smooth, the number of points required to approximate it may be quite large, and the exact position of the points somewhat unrelated to the contour itself. These issues have been tackled by the graphics community in the context of design, and we propose to use some of the resulting tools, particularly approximating B-splines. The resulting representation is compact and faithful to the original data for smooth or piecewise smooth contours, open or closed.

It is also very well suited for the detection of symmetries. Whereas it is easy to define symmetry between two infinite straight lines, the concept of symmetry between curves is harder to define: Rosenfeld [Rosenfeld, 1986] provides a lucid account of the differences between Blum's [Blum, 1967], Brooks' [Brooks, 1981], and Brady's [Asada and Brady, 1984] definitions, and a more recent paper by Ponce [Ponce, 1988] gives further comparisons. Here, we are interested not in local symmetries which provide skeletal shape primitives, but rather in symmetries which help to infer shape from contour: Nevatia and Ulupinar [Ulupinar and Nevatia, 1988] postulate that they are skewed and parallel.

These can be computed efficiently using our B-spline representation. The main advantages are the low computational complexity ($O(n^2)$, where n is the number of spline segments instead of the number of points) of the process and the stability of the results. Figure 14 shows an example of parallel symmetry detection using a quadratic B-spline approximation starting from the two digital curves displayed in figure 14(a).

As an application, for the very specific case of a torus, the detection of parallel symmetries allows us to infer the 3-D orientation of the object in a much simpler fashion than proposed in [Ponce and Kriegman, 1989], as shown on figure 15.

2.5 MATCHING

Object recognition involves identifying a correspondence between part of an image and a particular view of a known object. This requires matching the image against stored object models to determine if any of the models could produce a portion of the image. We have actively promoted the idea that higher level features organized in graphs are the key to recognition in the presence of occlusion and photometric variations [Nevatia and Price, 1982; Medioni and Nevatia, 1984; Fan *et al.*, 1989]. Recently, we have addressed the issues involved in recognizing objects in a cluttered environment when the number of models is large. We have been able to show excellent results for the recognition of flat objects under affine transform [Stein and Medioni, 1990a], and of 3-D objects given 3-D data [Stein and Medioni, 1990b]. The keys to our approach are

- a redundant representation
- Gray code to measure semantic difference
- hash tables for fast retrieval

• automatic acquisition of models

For the problem of recognition of multiple flat objects in a cluttered environment from an arbitrary viewpoint [Stein and Medioni, 1990a], the models are acquired automatically and initially approximated by polygons with multiple line tolerances for robustness. Groups of consecutive linear segments (super segments) are then quantized with a Gray code and entered into a hash table. This provides the essential mechanism for indexing and fast retrieval. Once the data base of all models is built, the recognition proceeds by segmenting the scene into a polygonal approximation; the Gray code for each super segment retrieves model hypotheses from the hash table. Hypotheses are clustered if they are mutually consistent, and represent the instance of a model. Finally, the estimate of the transformation is refined. This methodology allows us to recognize models in the presence of noise, occlusion, scale, rotation, translation and weak perspective. Unlike most of the current systems, its complexity grows as $O(kN)$ when N is the number of models, and $k \ll 1$.

An example of successful recognition is shown in figure 18 in the aerial image section.

For the recognition of 3-D objects from 3-D data, we use a data structure called a *splash*, which describes the variation of surface normals in a circular neighborhood of a point, encoded as a super segment. From then on, the matching methodology is identical to the 2-D case. The full details can be found in [Stein and Medioni, 1990b].

3 AERIAL IMAGE ANALYSIS

We have three projects for the analysis of images of aerial scenes including efforts to develop modules that exhibit high performance by themselves, the integration of modules into systems, and the formulation of a theory to define the underlying "visual abilities" required and useful for extraction of cultural features from images of aerial scenes:

- The focus of our work in the past has been the development of modules for detection and description of cultural (man-made) features present in aerial scenes such as the transportation network (fig. 16a,b) [Huertas *et al.*, 1990; Huertas *et al.*, 1989], building structures (fig. 17a,b,c) [Huertas and Nevatia, 1988b; Mohan and Nevatia, 1988; Mohan, 1989] and aircraft (fig. 18a,b,c) [Stein and Medioni, 1990a]. Below we give an example of a module for pier and ships detection from the image of a harbor complex.

These modules typically rely on perceptual grouping of primitive geometric features (lines, anti-parallel, junctions, portions of rectangles, etc) extracted from the images, to detect the objects. Modules for mobile objects such as aircraft and ships on the other hand, use models and rely on scale and rotation invariant matching techniques to detect the objects. Current work on 2-D and 3-D matching techniques is covered in detail in other papers in these proceedings [Stein and Medioni, 1990b] or recent

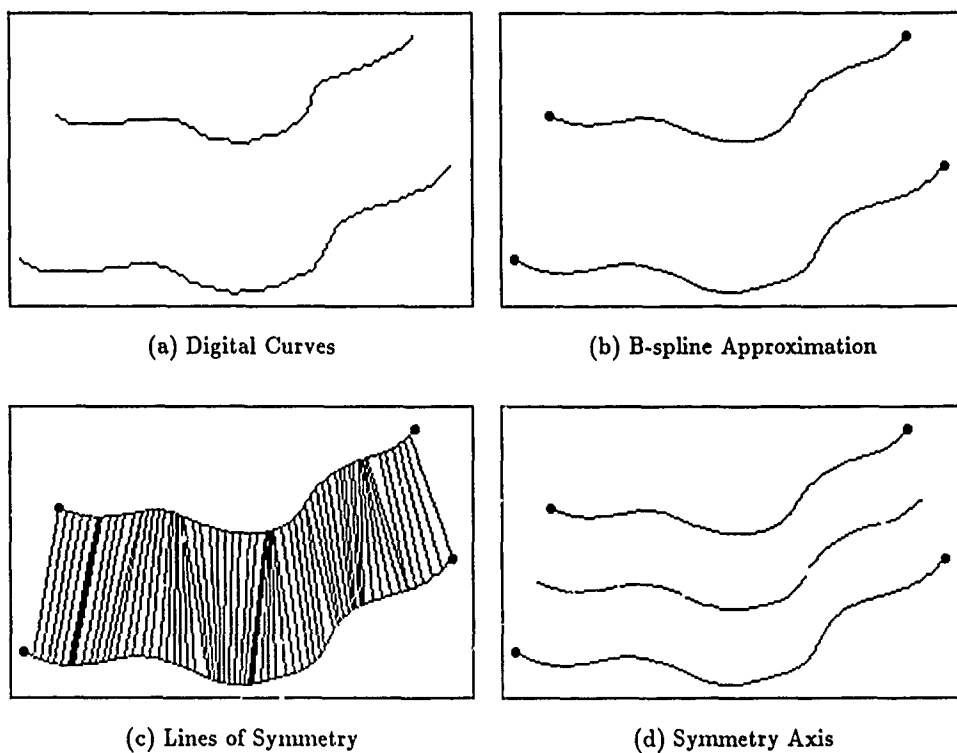


Figure 14: Detection of Elementary Parallel Symmetries

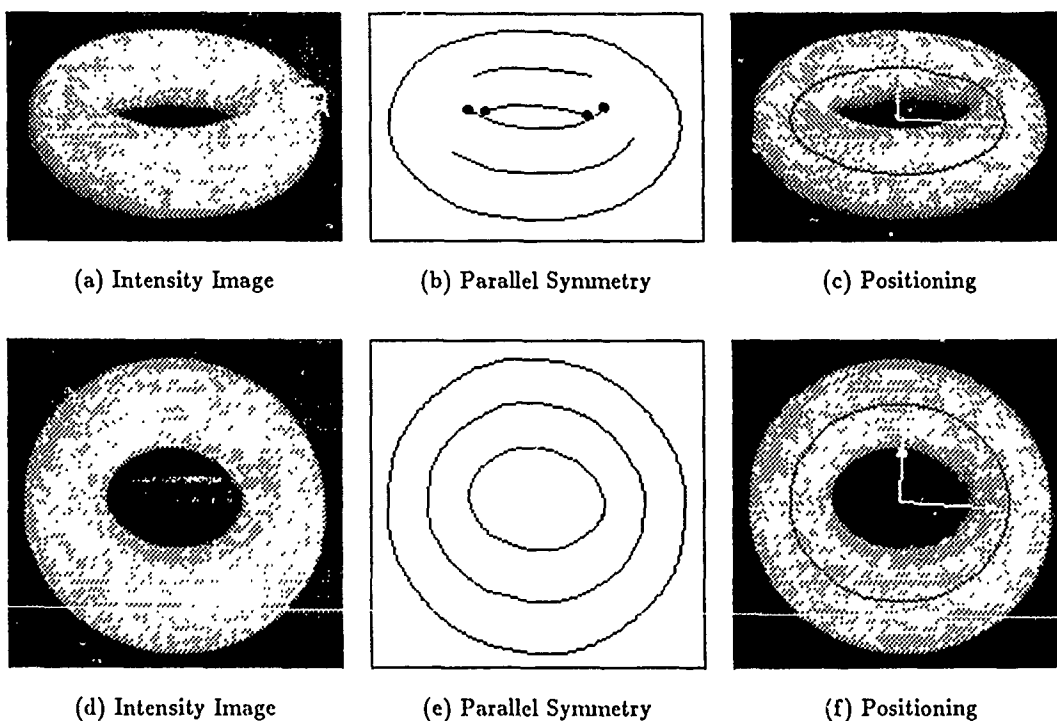
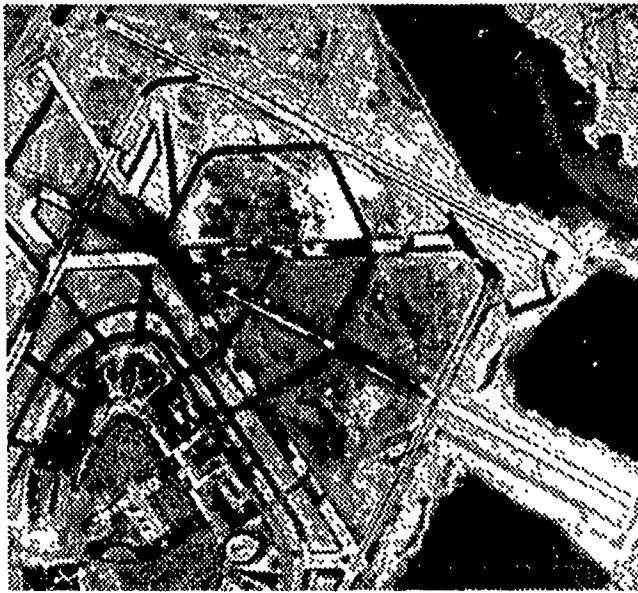
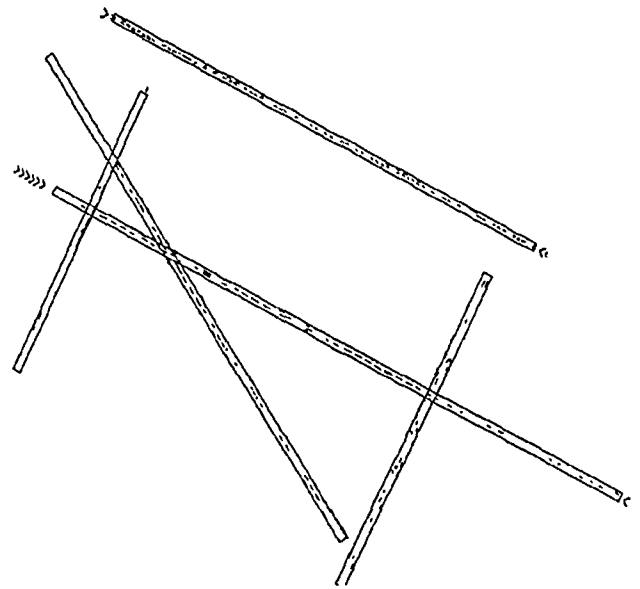


Figure 15: Positioning of a Torus



(a) JFK Airport



(b) JFK Runways

Figure 16: Runway detection module

conference papers [Stein and Medioni, 1990a]. Typically these methods are applied at a stage where we have a great deal of confidence that these objects are (or should be) present in the image. For instance, after detection of runways, taxiways, and buildings, we can then look for aircraft in the appropriate areas. These in turn, help reinforce the runway and taxiway hypotheses made as well as help determine the functionality of some of the buildings.

- A second portion of our work has concentrated on devising a system that manages the modules and integrates the results of the modules thus providing local and global context as well as higher level reasoning suitable for the description of an entire complex or scene. In the past we have concentrated in the domain of large commercial airports, and developed modules for detecting major structures. Now we are investigating the interaction of these modules. We hope to report on this work in a future paper.
- Our third project concentrates on the development of general techniques. These include devising a taxonomy of perceptual grouping operations and, the development of a language for describing tasks in terms of grouping operations. We expand on these topics in the following sections.

3.1 DEVELOPMENT OF GENERAL TECHNIQUES

We believe that a hierarchy of processing steps is the appropriate approach for aerial image understanding, where the levels of the hierarchy are chiefly determined by three factors:

1. The available sources of knowledge, both generic and domain specific. We know for instance that airport runways are straight (geometry), and that they must have standard markings (object specific) applied to the surfaces for safety and to aid pilots.
2. The available image resolution and quality. For example, it is more desirable to look for global features, such as harbor piers, at lower resolutions and then apply the model-to-feature matching to small portions of high resolution images to locate the ships (see below). Why? Because the pier areas are salient features, a collection of macro features arranged in some simple geometric fashion along the boundary of two distinct regions, land and water. The detection of ships, and perhaps their classification by type on the other hand, requires higher resolution and more symbolic processing.
3. Measurements and assertions as a function of scale. What can or should be measured at a given scale? Invariably we can get bogged down into considering everything possible at all scales, and build complex and massive data structures. However, this is often unreasonable for mapping and photointerpretation tasks where the image content and typical conditions quickly make some approaches unfeasible.

The characterization of such hierarchies is a central part of our work and involves two factors: the development of a formal language for describing the development of a grouping theory to support the "visual abilities" required to accomplish the tasks, and the development of a taxonomy of "visual abilities" required to accomplish the tasks. The development of a taxonomy of "visual abilities" required to accomplish the tasks is a central part of our work and involves two factors: the development of a formal language for describing the development of a grouping theory to support the "visual abilities" required to accomplish the tasks, and the development of a taxonomy of "visual abilities" required to accomplish the tasks.

terms of generalized classes of perceptual grouping operations that can be applied in parallel. Eventually the task descriptions should be given in terms of (or, compiled into) a sequence of alternating abstractions in the representation of the features and application of classes of grouping operations. We explore some of these ideas below using as an example the task to "detect pier areas and ships" from an image of a portion of a harbor aerial scene.

Most of this work would fit at the "middle-level" level of perception. The "connection" with the lower levels of processing, is reflected by the fact that the grouping processes are more "non-purposive", and thus should be implemented to run in parallel. The connection to higher levels of processing (reasoning about segmented objects, where an object is a single, functionally identifiable 3-D object, as determined by the task at hand) is reflected by grouping processes that are more purposive, operate on increasingly abstract features, and are sequential in nature.

Our group has for a number of years developed methods and techniques involving perceptual organization. Groupings of near, parallel, collinear, co-curved, and symmetric features have been used to represent, segment and extract parts or whole objects from aerial images to images of office scenes. For an excellent reference on our most recent work see [Mohan, 1989].

In the recent past we have begun work towards the development of a taxonomy for grouping operations, and here we only introduce informally the notion of *grouping fields*, a general tool for describing mathematically the visual abilities that involve perceptual groupings of visual primitives closer to the lower and middle levels of perception. These are analogous to the ability that humans have to, presumably preattentively, acquire sensations that capture, nearly instantaneously, fundamental and basic geometric arrangements of image elements in a reflexive manner.

Briefly, the notion of a *grouping field* is analogous to force fields in nature. When a visual feature, due to its size, shape, or other property induce a perceptual grouping with other features in the field of view, we say that a *grouping field* exists around it. Conversely, any visual feature in the field of view generates a grouping field which is a function of the feature properties and can be influenced by the task at hand.

We believe that grouping fields will be useful in dealing with many of the problems pointed out in the recent past in previous work by [Lowe, 1985; Lowe and Binford, 1983; Lowe and Binford, 1982; Witkin and Tenenbaum, 1983; Stevens and Brookes, 1987] and others, attempting to derive computational approaches to perceptual organization abilities.

The combinatorial explosions that arise in attempting to establish relationships among low level features purely on the basis of attribute processing is a major problem. For photointerpretation tasks, at least, it seems that the way to avoid this is to explore the generality aspects top-down, that is, by describing what we want, say detect piers and ships, and with our own experienced knowledge of piers and ships, generate a task description that

include the perception landmarks (first, detect border between land and water region, then detect pier areas, next detect ships in the neighborhood of pier areas, last identify ships.)

3.2 AN EXAMPLE: ANALYSIS OF HARBOR COMPLEXES

In analyzing a harbor or port complex we want to be able to describe the buildings in the port facility, the transportation network around the facilities, and of course the pier areas and the ships in the area. In our example we concentrate on the piers and ships and the grouping fields and grouping operations that lead to the detection of the pier areas. We then briefly discuss ship detection and classification.

What do we need to know about port and harbor facilities to detect the piers and describe the ships? That the planning and design of port and harbor facilities is strongly dependent on the characteristics of the ships to be served, and the type of cargo to be handled [Wright and Ashford, 1989]. To eventually describe the scene completely we would have to know a lot of things about the ships: Main dimensions (length, beam, draft), cargo-carrying capacity, cargo-handling gear, types of cargo units, shape, hull strength and motion characteristics, mooring equipment, maneuverability, and so on.

To detect only the pier areas (where later we look for ships) we only need the upper bounds on ship dimensions and the image resolution. These parameters are easily available a-priori and chiefly determine the extent and strength of the grouping fields associated with the features. Let us define a some grouping classes useful for this task:

- Proximity-0D (Px0D): Groups nearby features without regard for the dimensions of the features. Each feature, whether a dot, a line, a ship, or another suitable represented group, generates a grouping field about its center of mass. The extent of the field (typically, circularly symmetric) is determined by field of view or by the task at hand as a function of image resolution. Intersecting fields form a group with the same extent and has a new center of mass. The strength of the field is proportional to the "mass" (a function of the complexity of the feature), and inversely proportional to the square of the distance from the feature's center of mass. A scaling resolution constant is introduced so that the same two features at two different resolutions attract each other with the same force.
- Proximity-1D (Px1D): Groups nearby features where a 1D attribute is dominant and can be used to constraint membership. The strength of the field in this case would be proportional, and a function of the attribute.
- Proximity-ND (PxND): Groups nearby features with ND attributes. Each attribute requires one layer.
- Parallelism with overlap (PlwO): Groups features that are parallel to each other with respect to their

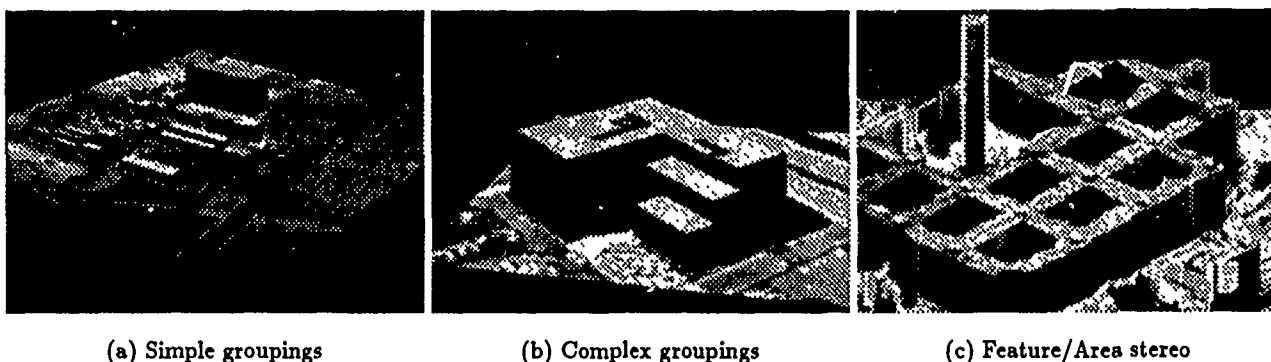


Figure 17: Building detection modules

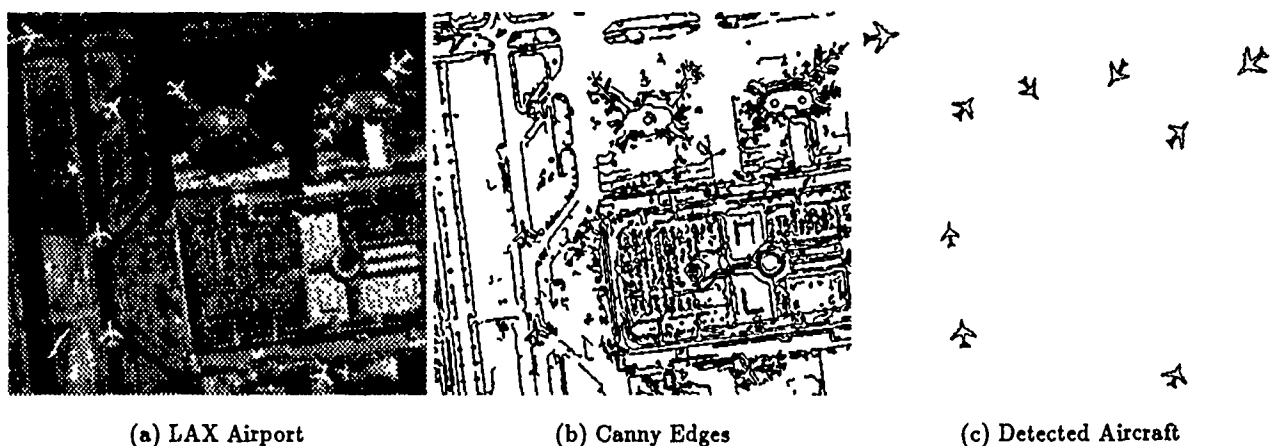


Figure 18: 2-D Matcher applied to image edges for aircraft detection

dominant orientations. Each allowed orientation determines a layer where the fields of each feature having that orientation is active. Intersecting fields give for each orientation all the features parallel to a given feature. The fields themselves have an elliptical shape with its minor axis equivalent to the length of the feature in the dominant orientation, and its major axis equivalent to the extent of the field of view or, constrained by the task at hand. Note that allowing for angle tolerances is equivalent to the intersection of fields across field layers.

- Parallelism with no overlap (PlwnO): The same as above with circularly symmetric fields.
- Collinearity-0D (Co0D): Groups three or more features without regard for the spatial extent of the feature. Any two of the three features determine the extent of the grouping field, typically an ellipse with high eccentricity, centered about the center of mass of the feature. The eccentricity determines the allowed tolerance in collinearity, and the extent of the field is equivalent to the extent of the field of view. The orientation of the two selected features

determines a layer for field intersection. The steps in a ladder have Co0D.

- Collinearity-1D (Co1D): Groups two or more features with respect to their dominant orientation. Each feature determines the extent its GF, also an ellipse with high eccentricity. The eccentricity determines the allowed tolerance in collinearity, and the extent of the field is equivalent to the extent of the field of view, or constrained by the task at hand. The orientation of each feature determines a layer for field intersection. The fragments of an airport runway have Co1D.

Let us now apply two of these definitions to our pier example:

Figure 19 shows an image of a portion of the U.S. Navy facilities in San Diego. We know that we should expect to see mostly military ships that may require long term docking, thus allowing for double or triple docking. We know the image resolution and the approximate ship dimensions, thus we know the minimum size of the piers. The following the levels of the desired task:

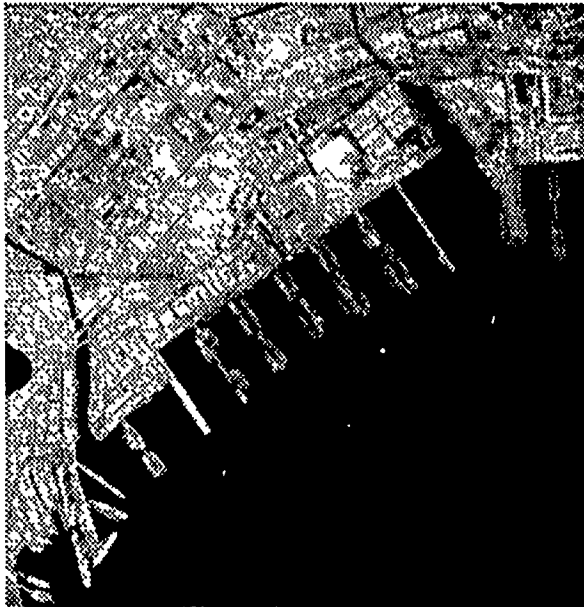


Figure 19: U.S. Navy Facility (512x512 image)

- 0: Analyze Harbor Scene.
- 1: Detect and classify buildings.
- 1: Detect and classify access roads.
- 1: Detect and classify ships.
- 2: Detect ships.
- 2: Locate ship repair/construction areas.
- 3: Locate ships.
- 4: Classify ships.
- 2: Locate Pier areas.
- 3: Locate boundary between land and water.
- 3: Locate "land" structures in water.
- 3: Detect pier areas.
- 3: Locate ships.
- 4: Classify ships.
- 3: Describe ships
- 2: Describe piers.
- 1: Describe piers and ships by class.
- 0: Describe harbor scene.

We now describe the task at level 2, Locate Pier Areas:

Locate Boundary between Land and Water: We detect the boundary between land and water regions automatically using our implementation of [Ohlander *et al.*, 1978]. In this example we arbitrarily selected the largest region to represent the water region. Next we approximate these boundary by piecewise linear segments (thick lines in fig. 20) using LINEAR, our implementation of [Nevatia and Babu, 1980].

Locate "land" Structures in Water: Contrary to many natural structures on the shores, man-made structures appear highly geometric. We expect that most piers appear as linear structures attached to the shore, and in the water. Their linearity indicates that the piers or portions of piers should be characterized by anti-parallel pairs of segments of opposing contrast [Nevatia and Babu, 1980], or *apars* for short. Ships are typically docked parallel and adjacent to the piers. We

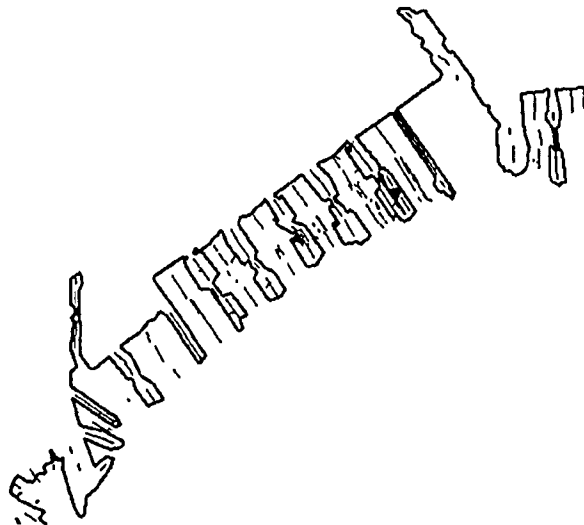


Figure 20: Line Segments and anti-parallel

then expect that most of the line segments corresponding to sides of piers, sides of ships, shadows, and so on in the neighborhood of the piers would result in many apars. The constraint on the range of separations between pair of segments (equivalent to the width of the resulting apar) is a function of image resolution and ship dimensions. The apars in our example are shown as thin lines in fig. 20 obtained also using LINEAR.

Detect Pier Areas: The apars are easily classified into land or water apar according to the detected water region. Subsequent processing operates on the land apars only. Next, we apply Px0D grouping to the land apars. The extent of the fields is task-dependent and does not have to be precisely determined. At the resolution in our example (about 8 meters per pixel), the fields radii is roughly equivalent to a pier width plus the width of three destroyers on both sides of the piers, or about 16 pixels.

These fields (fig. 21) occupy a single layer. Each field intersection operation shifts the center of mass of the group, however the field associated with the group has the same properties as the individual apar fields. We then select the groups so that apar membership is exclusive by extracting the groups in order of decreasing mass (number of apars). The resulting groups (fields) represent potential pier fragments (fig. 22.)

At any resolution, we expect that the lines be fragmented and incomplete, due to inefficiency in the line detection process or due to real structures in the image. Thus, we expect that the resulting groups represent pier fragments rather than complete pier areas. Since we expect the pier sections to be straight, the next step calls for collinearity grouping to join possible fragmented pier areas. Note that the groups in fig. 22 are easily perceived as being collinear.

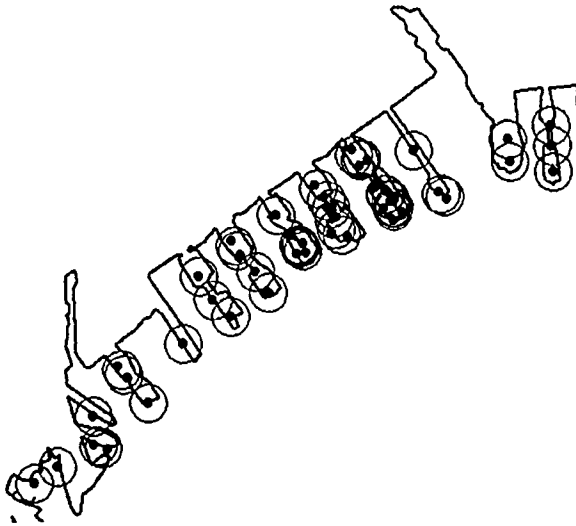


Figure 21: Px0D - Proximity grouping fields

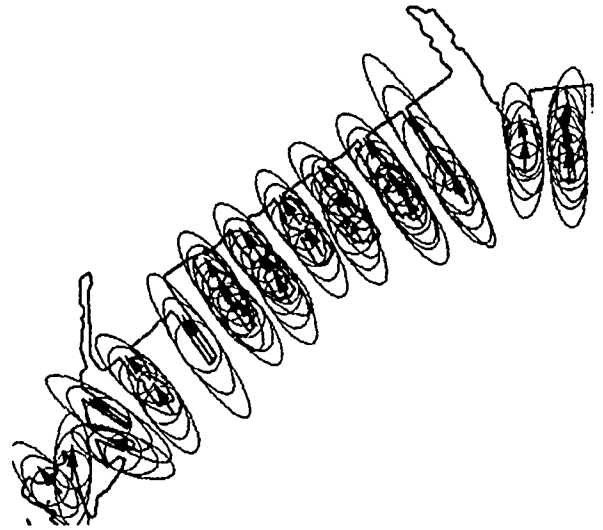


Figure 23: Potential pier fragments and Co1D fields

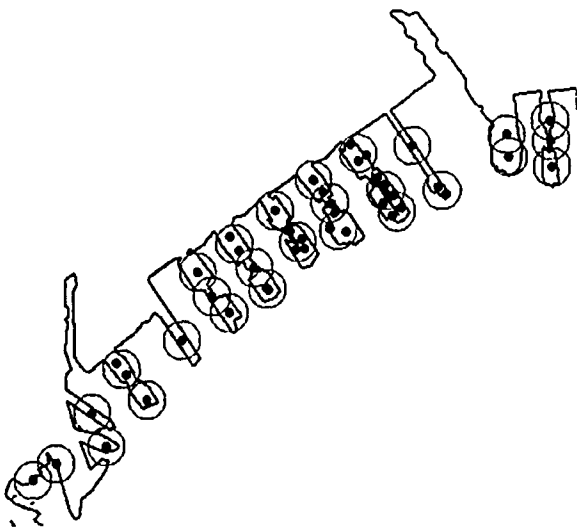


Figure 22: Selected proximity groups

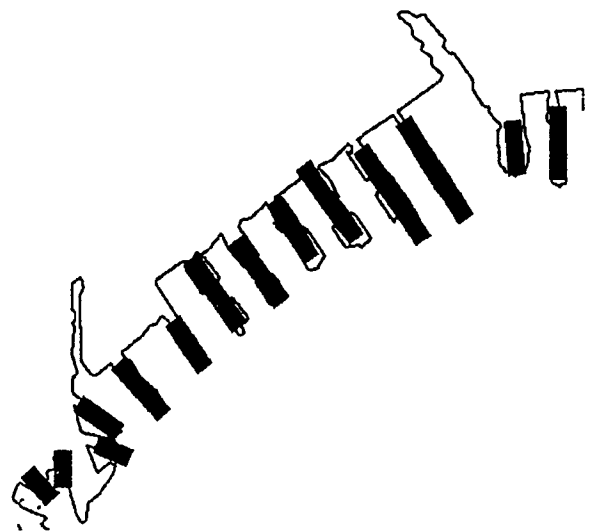


Figure 24: Detected Pier Areas (low resolution)

We choose to represent the groups of apars by apars as well, having a length and width equal to the diameter of the final field. The orientation of the apar is given by the dominant orientation (the largest peak in the length-weighted histogram of the orientation) of the apars in the group (see arrows in fig 23.)

Next we apply Co1D to the pier area fragments. The longest piers are about three times the length of a destroyer thus we allow the extent of the elliptic fields (see fig. 23) to be up to three times the apars, and have a width equivalent to the apar width (or group radius).

The result of the grouping is then represented, again

by apars, which in turn represent potential pier areas (see fig. 24). These are described by their approximate length and position, and are used to extract image windows from a high resolution image of the scene where we look for ships.

Locate ships: We have performed some preliminary experiments to detect the ships in the high resolution windows using the same matching technique we have used in the past to detect aircraft [Stein and Medioni, 1990b]. One of these windows is shown in figure 25a, and the corresponding adaptively smoothed [Chen, 1989]

boundaries in figure 25b. Three coarse-to-fine models of a single and a double destroyer group were matched against these edges to obtain the detected ships in figures 25c,d.

Classify ships: We consider our ship detection results preliminary. The simplicity of the ships shape becomes a disadvantage to the matching technique. The double ship configurations are easier to match for the same reason. For ship identification better ship boundaries are required. We plan to apply a technique for boundary refinement using B-snakes [Menet *et al.*, 1990]. The matching technique can then be applied with finer models for more accurate ship classification. Other alternatives for ship detection include stereo processing of these high resolution windows with a area/feature based technique [Cochran and Medioni, 1989a], also followed by boundary refinement and 2D matching.

3.3 FUTURE WORK

We plan to spend some time in the development of the notion of grouping fields and grouping classes, in conjunction with the development of a task description language for mapping and photointerpretation tasks. To be useful we believe that grouping field generation and manipulation should be amenable for parallel implementation, and hopefully can be extended to other domains.

4 MOTION ANALYSIS

We have a number of projects in the analysis of sequences of images including analysis of closely spaced images, feature based analysis, motion estimation techniques, and navigation using recognition of visual features. Autonomous navigation provides the context for much of the work, though the techniques have a much broader utility.

Motion analysis using feature point analysis techniques and multiple frames forms the central focus of our work. This approach involves extracting a set of consistent features from a sequence of images, finding the corresponding features in consecutive frames, and finally computing the three-dimensional motion based on the correspondences, which also provides an estimate of the structure of the moving objects or scene. These are often described separately or as sequential operations, but integration into a single system and feedback to earlier processing is a major part of the work.

Our effort includes several separate and related projects including: analysis of closely spaced images (spatio-temporal analysis) using features such as lines, corners, and regions to extract three-dimensional structure information, matching edge based contours in a sequence of images, integrating several feature detection and matching techniques to derive three-dimensional motion and structure estimates, study of the formulation of the motion estimation problem, detection of moving objects in a scene with a moving observer, and the visual guidance of a mobile robot. This overview discusses the current status of the research in these areas. Some of these are covered in more detail in other papers in this proceedings or in other recent conference papers.

4.1 SPATIO-TEMPORAL ANALYSIS

The goal of our work in spatio-temporal analysis is to generate a dense optic flow map from a motion sequence. Because of the sparseness of 0D features (corners) or 1D features (curves), we feel 2D features (regions) are more likely to produce dense motion estimates.

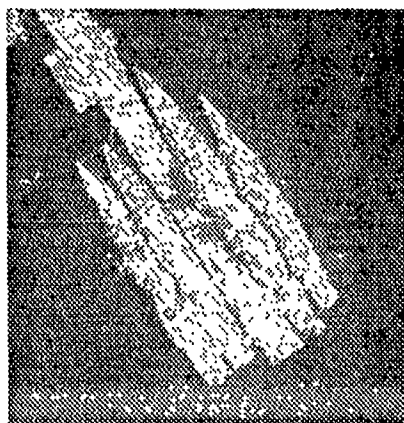
Early work in spatio-temporal analysis includes that of [Bolles *et al.*, 1987]. We began our work with [Peng and Medioni, 1988; Peng and Medioni, 1989], which extracts paths in slices taken from image sequences in the temporal direction (i.e. paths of an object point through time and space). The slopes of the paths carry important motion information, the flow normal to the contour can be resolved by combining path slopes extracted from different slicing orientations. The drawback of this method is that we only compute motion estimates on contour points.

If we examine the slices more carefully, some pairs of paths serve as the non-parallel sides of trapezoidal regions. Each such region corresponds to a collection of chords of a moving object seen in each image in the sequence. If we assume that the velocity changes smoothly between two paths (i.e. between two points on an object), we generate flow values for all pixels in the region by interpolation.

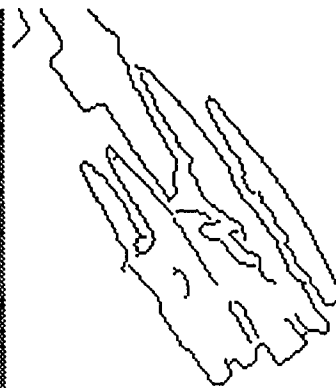
Assuming the motion in the scene is approximated by piecewise translational motion along the axis of the camera, and the focus of expansion (FOE) position is given, the optical flow direction of each image element can be determined. If slices are cut at the FOE along the direction radiating from the FOE and image points are matched in the slice, the match disparities are then the magnitude of the velocity. This, when combined with the interpolation, produces a dense optic flow map.

Since the spatio-temporal images are registered in a Cartesian coordinate system, cutting radial slices is equivalent to transforming the images into a polar coordinate system. This causes resolution problems: if we slice the sequence densely enough so that all pixels far from the FOE are in at least one slice then pixels close to the FOE are included in many slices.

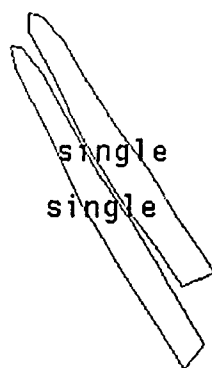
We devised a parallel algorithm to approximate the complete radial slicing, which simplifies this data access problem. We only take slices at each pixel along the four directions: horizontal, vertical, 45° , and -45° . Using the interpolation step mentioned above, each pixel would have at most four estimates of the velocity components along different directions. Using the method presented earlier in [Peng and Medioni, 1988; Peng and Medioni, 1989], the normal velocity of the pixel is recovered. With both the motion direction (from the FOE position) and normal velocity (from the slice analysis), we are able to compute the velocity of the pixel. From our experiments, the results from both approaches are very similar. Results of using this technique are shown in figure 26 and 27 showing one frame of a sequence, the computed velocity for the optical flow and the typical optical flow direction diagram.



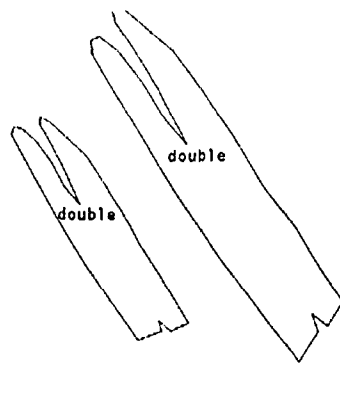
(a) Ships 160x160 image window



(b) Canny edges

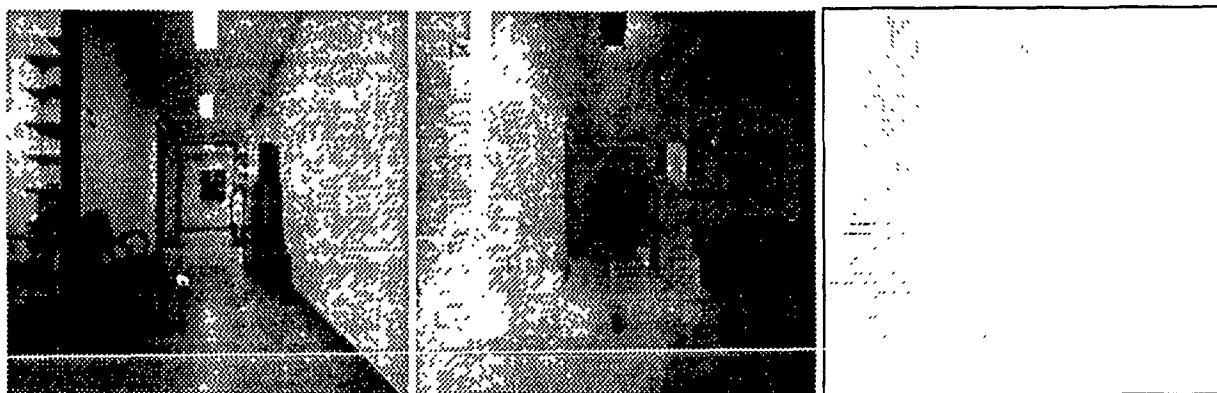


(c) Single ship match



(d) Double ship match

Figure 25: Fast 2D model-based matcher applied to edges of ships



(a) First Frame

(b) Velocity

(c) Needle Diagram

Figure 26: SRI Sequence: Hallway

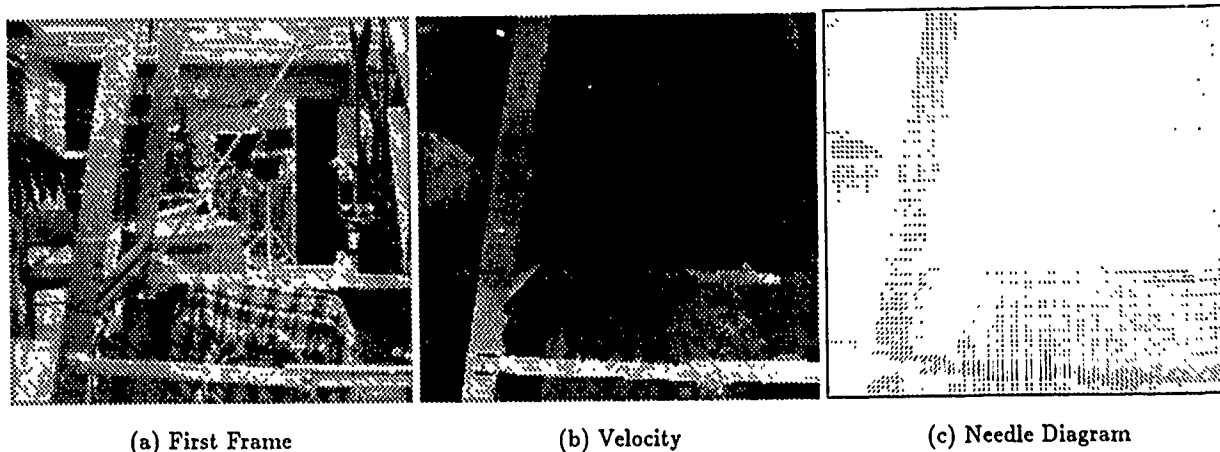


Figure 27: SRI Sequence: Zoom

4.2 FEATURE-BASED MOTION CORRESPONDENCE

This contour-based matching technique is developed from our previously reported work [Gazit and Medioni, 1988; Gazit and Medioni, 1989]. The major changes are that multiple frame matches need not extend through the entire sequence of images, thus allowing for occlusion and (re)appearance of points midway through the sequence; and the use of *neighborhood* and *length* to distinguish between correct and incorrect matches. These changes have resulted in a significant improvement in the quality of the matches.

A brief description of the contour matching method is: A *super-segment* is an object described both as a list of connected edgels and a list of connected line-segments (that approximate the edgel contour). The algorithm tries to match sections of super-segments. Since a single object may correspond to several different super-segments and a single super-segment may include more than one object, the problem is to identify the matching super-segment *sections*. We base our initial matching criterion only on *shape similarity* and *proximity* (with a maximum allowable disparity). An initial approximation is found by first matching the line-segments and combining matches along each super-segment. Next we compute the section matches themselves. In order to find appropriate matching sections, we break the line-segment approximations used in the previous stage into arbitrary small sections and match them (along the possibly matching section) by maximizing the similarity between the matching sections as well as the length (in points) of the matching sections. The result is a very large set of matches, the great majority of which are spurious. The main thrust of the work is in how to deal with these spurious matches.

Our solution to distinguish between incorrect and correct matches is based on the assumption that correct matches will usually either be *long* or will have *approving neighbors*, which are neighbor matches representing a similar motion. The neighborhood size should ide-

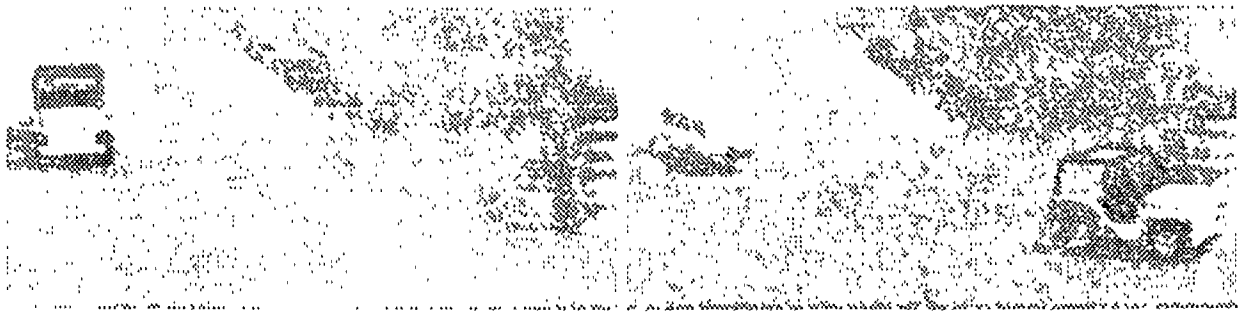
ally depend on object size, but since this step comes before object segmentation, we instead use a fixed fraction of the image size. Each match is assigned a *approving length* score which is a combination of the total length of supporting neighboring matches and their number; a *non-approving length* score computed from the non-supporting neighbor matches; and a *shape similarity* score. Using these three measures together allows us to detect incorrect matches in most of the cases, since they are short, have little neighborhood support and a lot of neighborhood rejection, as they represent an inconsistent motion. A notable exception to this occurs with straight line contours, which are easy to detect and for which we have a partial solution, and repetitive structures.

We apply this algorithm hierarchically for different scales for better performance [Gazit and Medioni, 1989]. We also combine pairwise matches into multiple-frame matches by tracking the matching sections through the sequence. If section P_1 in frame 1 matches section P_2 in frame 2, Q_2 in frame 2 matches Q_3 in frame 3 and P_2 overlap Q_2 , we can compute a new match (R_1, R_2, R_3) corresponding to the overlapping part. This is applied to all frames in the image sequence.

The resulting section matches can be used for 3D motion estimation or motion segmentation. We are currently working on the latter case.

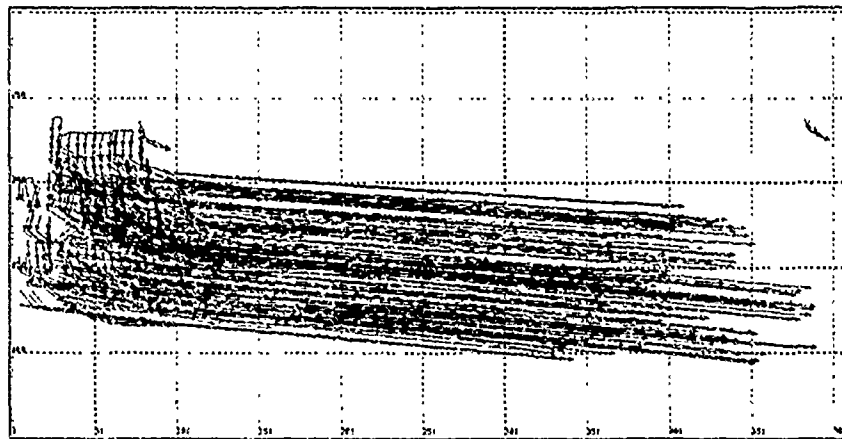
As an example, a sequence consisting of 10 250×512 frames of a toy jeep and train is given in figure 28. We only show the last two frames and the resulting multiple matches. Because the motion of the objects overlap, and also to allow for better visibility, we manually removed the background matches and separated the matches corresponding to the train and those corresponding to the jeep.

In this scene the camera is stationary, but both the jeep and the train are moving. This is a difficult scene as the motion is very large (disparity ranges from 0 to 150 pixels) and the scene contains occlusion.

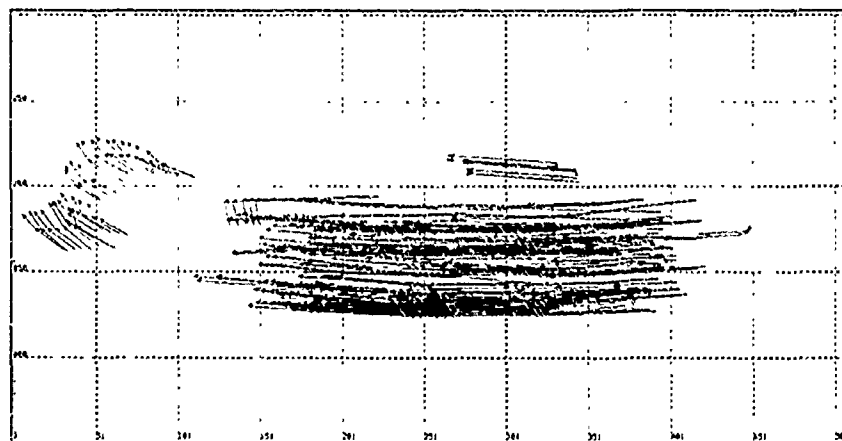


(a) First Frame

(b) Last Frame



(c) Multiple Matches of the jeep



(d) Multiple Matches of the train

Figure 28: Jeep and train image - Multiple Matches

4.3 DETECTING MOVING OBJECTS FROM A MOVING PLATFORM

Detecting moving objects from a moving platform is a difficult problem, because the observer motion causes stationary objects to appear to move. Thus, we must separate genuine motion from the apparent motion of the stationary environment. We have developed a system that successfully detects moving objects within a sequence of real images taken from an observation vehicle traveling along a road. Unlike other systems, ours does not require densely-sampled imagery, meaning that objects can move many pixels per frame with no detrimental effects. Nor does the system rely on critical parameter settings. It is computationally efficient, and highly suited to parallel implementation. It requires no object matching or recognition, and can thus detect moving objects that are partially occluded or that are camouflaged.

When an observer moves in a straight line, toward a distant point in space, stationary objects in the environment appear to move along paths radiating from that point. The point from which the paths radiate is called the focus of expansion (FOE). We assume that the FOE, and camera orientation, are relatively stable between successive images (i.e., the observer must not sharply turn or tilt between images).

To simplify the problem, we first perform a Complex Logarithmic Mapping (CLM) as suggested by [Cavanaugh, 1978; Weiman and Chaikin, 1979; Jain, 1984]. This converts the problem from one of detecting a complex motion along both the X and Y axes, to one of detecting motion along an angular axis, with stationary objects moving along the other axis.

To detect the angular motion in CLM space, we have developed a novel "moving-edge detector," which operates on successive images and produces a map containing all pixels on the edge of regions that are moving relative to the stationary background. This map is then thresholded to produce detected movement. Preliminary results indicate that, once the threshold is raised high enough to eliminate false alarms, it can be increased by a factor of five and still properly detect moving objects.

The resulting detected movement is then transformed back into the rectangular reference frame, and overlaid upon the original image to highlight the detected objects. The results are presented in more detail in another paper in these proceedings [Frazier and Nevatia, 1990]. This technique depends heavily on the correct computation of the FOE and very loosely on the movement threshold value.

4.4 MOTION ESTIMATION

At USC, we continue an exploration of the multiframe structure from motion problem using feature matches. This work assumes a central projection pinhole camera with no smoothness assumptions imposed concerning object surfaces. The use of multiple (as opposed to two) frames is desirable for several reasons:

- to increase the robustness of the solution,
- to allow recovery of structure/motion with fewer features being tracked, and

- to allow estimation of "higher order derivatives" of the motion.

In our recent work, we have developed and implemented two algorithms to solve the SFM problem that make different assumptions concerning smoothness and type of motion. The first is a closed form algorithm that models the relative motion between the camera and the object or environment as a uniform 3D acceleration. The second is an iterative algorithm that can recover arbitrary rigid transformations between frames. The closed form algorithm is currently being used to generate initial guesses for the iterative algorithm when the rotation is known to be small.

The two algorithms share some characteristics: Both assume that features are matched through at least three frames. The image plane position of each feature is modeled as having a bivariate Gaussian error distribution, with the error coefficients provided as input. Although the algorithms are developed using point features, they can process both point and line features. A given feature is not required to be visible in every frame, so the algorithms can process features that become (un)occluded during the image sequence. As output, both algorithms generate the 3D location of each feature in each frame, along with the motion parameters.

The closed form algorithm models the motion as a uniform 3D acceleration. It minimizes a norm that is closely related to the maximum-likelihood image plane error norm subject to the constraint that the mean inter-frame displacement must equal one. Under this formulation, the 3D point positions are linear functions of the motion parameters, and the motion parameters can be determined by solving a small eigenvalue problem. The computational complexity of the algorithm is linear in the number of features being tracked times the number of frames and takes about 0.43 seconds on a Sun 3/280 for 8 points in 11 frames.

The iterative algorithm solves the SFM problem as an unconstrained minimization problem. The function to be minimized consists of 3 (classes of) terms:

1. the image plane error or a more or less convex approximation to it,
2. terms which bias the motion to be chronogeneous or some subclass of chronogeneous motion, and
3. a term which imposes a specific scale on the solution.

The exact set of terms to be used is still under investigation. Seemingly minor changes in the form of a term may dramatically alter the convergence properties of the algorithm. The algorithm is currently very slow because analytic derivatives have not been programmed, and a quasi-Newton method with a finite difference gradient is being used to do the optimization.

4.5 INTEGRATED SYSTEM FOR MOTION

We have continued to develop and use an integrated system for testing each of the subsystems of the motion analysis system (segmentation, feature extraction and matching, motion estimation, motion feedback to matching and coordination). The results of each subsystem is

saved in a single data structure and the coordination module controls exchange of information between subsystems. The integrated system is now being used to generate a rough description of three-dimensional structure of the environment, using region-based matches refined by corner matches over multiple frames. This work is described in more detail in these proceedings in [Kim and Price, 1990].

Feature matching is done in a coarse-to-fine manner to reduce search space and enhance stability. Corner-based matching for a region is guided by the motion computed for the centers of mass of the matched regions and by the constraint that matching corners are on the same region. This allows large disparities between images and different motions for each of the regions. Corner-based matching is performed both in the forward and reverse directions to decrease errors in matching.

We have developed a translation dominant motion analysis system as an additional feature of the general motion analysis system. The basic assumptions are that each object in the scene is undergoing a translation dominant motion and that an object may (or may not) be in coherent motion with some of the others. An approximate FOE (focus of expansion) using a LMSE (least mean square error) estimation and motion parameters are estimated for each region and then depth is computed for the corners of the region. Each computed result is associated with a reliability factor, which is a measure of the closeness to the computed motion to a translational motion. Regions with a high reliability are given high priorities in the analysis and their results act as a guide in the analysis of the less reliable regions by giving some constraints to the motion parameters.

This motion analysis system was tested for two real image sequences. A camera is moving straight along a hallway in one of them, and in the other sequence, a car is moving from the right side of the image to the other end. With a reasonable amount of noise, we could obtain an approximate environmental depth map for most of the important regions in the scene. Depth maps with region-corner matchings are shown in [Kim and Price, 1990] elsewhere in these proceedings.

Experiments show some weak points for this system. First, the use of the FOE analysis for general motion (translation + rotation) is sensitive to noise and thus the computed motion parameters are numerically unstable. In the case of translation dominant motion, an accurate estimation of FOE is essential for reliable results. Second, information of depth is lost along a smooth boundary even when it forms a great part of a region since fine structure is determined by corner matchings.

We will continue to add more features to our integrated system. First, we plan to add more feedback links within the integrated system so that an erroneous operation of early stages is detected and corrected by monitoring the results of later stages. This way, motion analysis is done as a part of a cooperative process rather than an isolated stage of a sequential process. Second, our system is to be expanded such that it can handle dynamic data and algorithms. The system would decide to continue to analyze intermediate frames in a tree

structure if necessary and would choose which algorithm to use, according to the characteristics of the image sequence. Finally, we intend to strengthen robustness to work well with reasonable amounts of noise for general motion analysis.

4.6 MOBILE PLATFORM

We recently acquired a Denning mobile robot for both indoor and outdoor experimentation. The initial phase of experimentation has dealt with basic control and navigation issues but the goals include visual feature navigation and a platform for testing our other motion algorithms. We do not intend to work on real-time (high speed) control, which would be possible with additional special purpose computers, but to develop high-performance analysis algorithms. The initial effort has included:

- implementing an obstacle avoidance routine using the range data provided by the 24 ultrasonic sensors of the robot, and
- building a simple planner allowing the robot to navigate indoors.

An obstacle in front of or on the sides of the robot is detected by checking the ultrasonic sensors in near the direction of motion. If there is an obstacle, the robot turns toward the direction of the first sensor where the path is clear. This is intended as a low-level survival process rather than a major navigational tool.

The map of the robot world is represented by a hierarchical data structure that includes buildings which are defined by a set of floors. Each floor has hall-ways, a set of rooms and a set of walls. Each wall may include doors.

In the first phase, the robot is assigned to navigate in the hall-way of a floor. The ideal trajectory is the mid-line between the two walls of the hall-way. The planner first computes a list of the axis of symmetry of each hall-way path. Each axis is limited to the common part of the pair of walls, must be inside the external polygon of the hall-way, but not inside any of its internal polygons. A merging step produces the axes shown in dashed lines on figure 29.

From the extremes and the intersections points of these axes, a graph of trajectory control points is constructed. The path of the robot, shown by thick black circles and lines on the figure, from its current location toward a goal door is then computed from the graph representation. Finally, the list of path control points is given to the navigation routine that orients the robot toward the next path control point, unless the robot is bypassing an obstacle. We will be incorporating visual object recognition into the navigation system but will continue to use this low-level guidance and sonar obstacle avoidance between visual control points.

5 PARALLEL PROCESSING

As shown in the previous sections, we are making good progress in solving some difficult image understanding problems. However, one major obstacle remains in applying our methods in practice, namely that of processing speed. Our algorithms, when run on a conventional

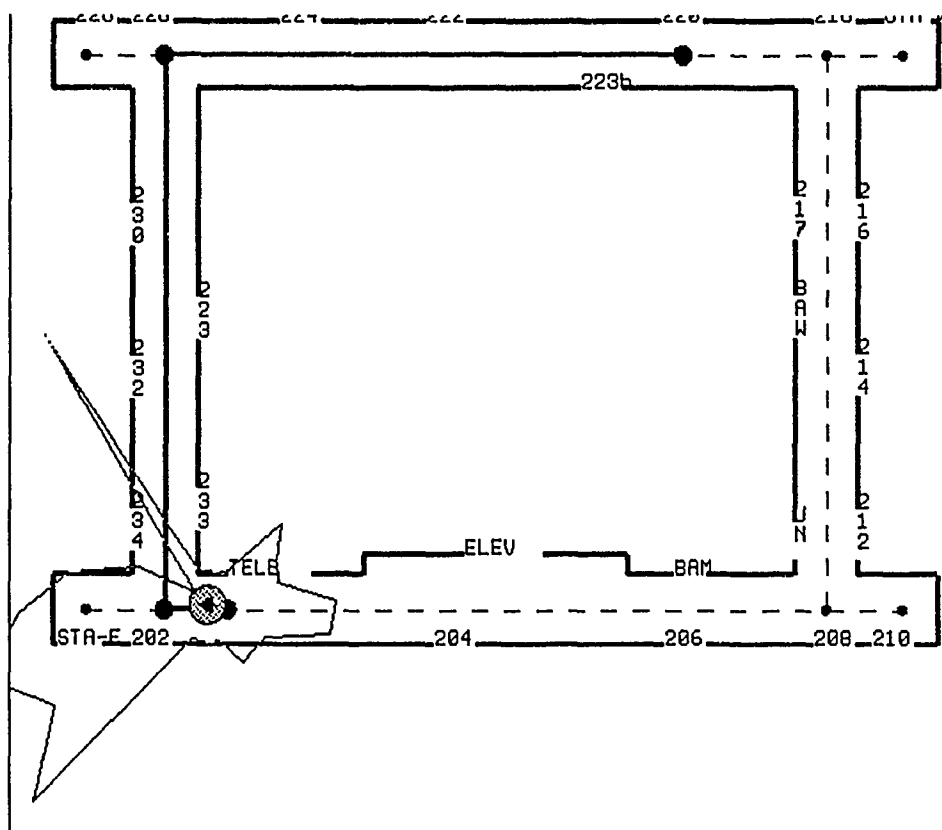


Figure 29: Floor map, Symmetry axis, Path and robot with real sonar data.

serial computer (such as a Symbolics 3600 or a Sun 3 or 4 series) can take several minutes or even hours to complete. We believe that this long execution time and its related computational complexity are inherent in the solution to the problems and hence we must devise ways of applying additional computing power to our algorithms. This naturally leads to the study of parallel computation.

There has been significant recent activity in applying parallel processing to image understanding problems. However, much of this activity focuses on numerical computations applied to iconic data structures. While such computations are necessary and useful, they are not nearly sufficient. Our approach to image understanding is firmly based on use of symbolic representations and symbolic computations. Parallelizing such computations is significantly more complex than for iconic, numerical computations and therefore, is the focus of our parallel processing research.

We have chosen some specific and representative medium and high level image understanding algorithms that we have found to be of general utility and are study-

ing their mapping onto suitable parallel architectures. Our goal is not only to map these specific algorithms, but also to learn how to parallelize classes of symbolic algorithms. One specific algorithm we have focused on is a "relaxation labelling" algorithm [Medioni and Nevatia, 1984]. We have found this algorithm to be useful in a variety of tasks in our work at USC; relaxation labelling has also been used by many other researchers elsewhere [Rosenfeld *et al.*, 1976].

We have obtained several efficient parallel implementations of discrete relaxation techniques on a class of parallel architectures [Lin and Prasanna Kumar, 1990]. Using these approaches, stereo matching and other labeling problems can be solved. First, a faster sequential algorithm compared to traditional approaches for discrete relaxation is developed. This algorithm is then parallelized and mapped onto a bus-connected parallel architecture. This mapping leads to a parallel execution time of $O(nm)$ using nm processors for consistently labeling n objects with m labels. Two versions of this design are developed; one for special-purpose VLSI implementation and the other for general-purpose parallel

architectures. The stereo matching technique developed in [Medioni and Nevatia, 1984] can then be modified to lead to an efficient parallel implementation based on the proposed solution.

The usual approach to parallel processing is to choose a specific architecture (based on considerations of availability as well as suitability) and then attempt to map the given algorithm onto it. This often leads to complex implementations that are difficult to understand and put a severe burden on the programmer. In recent work, we are taking an alternative approach of using a flexible architecture where the architecture can be modified to suit the data flow requirements of the algorithm. Flexible architectures are becoming feasible design solutions as commercial processing elements that support parallel processing, such as the Transputer, are becoming available. Efficient parallel implementation can be achieved while maintaining the structure of the program much as it is for the serial implementation. That is, parallel efficiency can be obtained while maintaining algorithm simplicity and keeping the programmer burden low. We have succeeded in demonstrating this approach for the relaxation algorithm; this work is described more fully in [Reinhart and Nevatia, 1990]. In future work, we intend to examine more complex algorithms and complete systems with this approach.

In another project, we are studying processor-time tradeoffs. These are of fundamental importance in understanding the complexity and performance of parallel computations. Driven by technological limitations, hardware cost, and flexibility, several schemes have been proposed for implementing large size computations on parallel architectures of fixed-size, or on architectures having a reduced number of processors. The major goal of such schemes is to keep the number of processors (or the processing chip-area, if implemented in VLSI) independent of the problem size and subject only to hardware cost, and other practical considerations. Such considerations are particularly important for problems on digitized images. With increasing image resolution, a processor array for a 1024×1024 image with a fixed number of pixels, say 8, per processor requires more than 10^5 processors. Design and implementation of such large arrays may be prohibitive, in addition to dealing with I/O limitations, programming and testing methodologies. Furthermore, if this array is required to handle larger size images, say images of size 2048×2048 , then processor-time trade-offs must be addressed again.

Direct mapping of parallel techniques from a specific organization onto a smaller version of the same organization generally does not lead to linear processor-time trade-off. New techniques based on combining efficient parallel and sequential algorithms must be developed. We have considered several parallel architectures with a large memory and a reduced number of processors for parallel image computations [Alnuweiri and Prasanna Kumar, 1990a; Alnuweiri and Prasanna Kumar, 1990b]. The memory size is proportional to the image size. However, the number of processors can be varied over a wide range while maintaining processor-time optimal performance. Architectures considered include

the *reduced mesh of trees* (RMOT), *mesh-connected modules* (MCM), *linear arrays*, *two dimensional meshes*, *hypercubes*, and *shuffle-exchange networks*. An alternate cost-effective parallel architecture, designated window architecture, is proposed for image understanding applications [Lin, 1990]. This architecture consists of a small number of processors with mesh connections and a large external memory with simple processor-memory access scheme. Parallel solutions for several image understanding problems, such as image labeling, computing image transforms, computing geometric properties, image and stereo matching using high level primitives such as line segments, have been derived on this architecture [Lin, 1990].

6 ACKNOWLEDGEMENTS

This paper describes the work of many faculty staff and students. Besides contributing to research, they also provided pieces of text that are incorporated in this paper in some form or the other. In particular, we wish to acknowledge the contributions of:

Dr. J. Y. Cartoux, J.-S. Chen, Y. Chen, R. Chung, S. D. Cochran, W. Cole, W. Franzen, J. Frazier, S. Gazit, A. Huertas, Y. C. Kim, S. Menet, B. Parvin, Prof. V. K. Prasanna-Kumar, S.-L. Peng C. Reinhart, Dr. P. Saint-Marc, F. Stein, and F. Ulupinar.

References

- [Alnuweiri and Prasanna Kumar, 1990a] H. Alnuweiri and V.K. Prasanna Kumar. Optimal image algorithms on an orthogonally connected memory architecture. In *Proceedings of the International Conference on Pattern Recognition*, pages 350-355, Atlantic City, New Jersey, June 1990.
- [Alnuweiri and Prasanna Kumar, 1990b] H. Alnuweiri and V.K. Prasanna Kumar. Optimal image computations on reduced processor parallel architectures. *Parallel Architectures and Algorithms for Image Understanding*, 1990. Academic Press.
- [Asada and Brady, 1984] H. Asada and M. Brady. The curvature primal sketch. In *Proceedings of the IEEE Workshop on Computer Vision: Representation and Control*, pages 8-17, Annapolis, Maryland, May 1984.
- [Blum, 1967] H. Blum. *A Transformation for Extracting New Descriptors of Shape*. MIT Press, Cambridge, MA, 1967.
- [Bolles et al., 1987] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1:7-55, 1987.
- [Brooks, 1981] R. A. Brooks. Symbolic reasoning among 3-D models and 2-D images. *Artificial Intelligence*, 17:285-349, 1981.
- [Cavanaugh, 1978] P. Cavanaugh. Size and position invariance in the visual system. *Perception*, 7:167-177, 1978.

- [Chen, 1989] J.-S. Chen. *Accurate Edge Detection for Multiscale Processing*. PhD thesis, University of Southern California, 1989.
- [Cochran and Medioni, 1989a] S. D. Cochran and G. Medioni. Accurate surface description from binocular stereo. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, California, May 1989.
- [Cochran and Medioni, 1989b] S. D. Cochran and G. Medioni. Accurate surface description from binocular stereo. In *Proceedings of the Workshop in Interpretation of 3D Scenes*, pages 16-23, Austin, Texas, November 27-29 1989.
- [Drumheller and Poggio, 1986] M. Drumheller and T. Poggio. On parallel stereo. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1439-1448, San Francisco, California, April 1986.
- [Fan et al., 1989] T.-J. Fan, G. Medioni, and R. Nevatia. Recognizing 3-d objects using surface descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1140-1157, November 1989.
- [Ferrie and Levine, 1987] F.P. Ferrie and M.D. Levine. Integrating information from multiple views. In *Proceedings of the IEEE Workshop on Computer Vision*, pages 117-122, December 1987.
- [Frazier and Nevatia, 1990] J. Frazier and R. Nevatia. Detecting moving objects from a moving platform. In *Proceedings of the DARPA Image Understanding Workshop*, Pittsburgh, Pennsylvania, September 1990.
- [Fua and Leclerc, 1988] P. Fua and Y. G. Leclerc. Model driven edge detection. In *Proceedings of the DARPA Image Understanding Workshop*, volume 2, pages 1016-1021, Cambridge, Massachusetts, April 1988.
- [Gazit and Medioni, 1988] S. L. Gazit and G. Medioni. Contour correspondences in dynamic imagery. In *Proceedings of the DARPA Image Understanding Workshop*, pages 423-432, Boston, Massachusetts, April 1988. Morgan Kaufmann Publishers, Inc.
- [Gazit and Medioni, 1989] S. L. Gazit and G. Medioni. Multi-scale contour matching in a motion sequence. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, California, May 1989. Morgan Kaufmann Publishers, Inc.
- [Hoff and Ahuja, 1989] W. Hoff and N. Ahuja. Surfaces from stereo: Integrating feature matching, disparity estimation, and contour detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(2):121-136, February 1989.
- [Huertas and Nevatia, 1988a] A. Huertas and R. Nevatia. Detecting buildings in aerial images. *Computer Vision, Graphics, and Image Processing*, 41(2):131-152, 1988.
- [Huertas and Nevatia, 1988b] A. Huertas and R. Nevatia. Detecting buildings in aerial images. *Computer Vision, Graphics, and Image Processing*, 41(2):131-152, February 1988.
- [Huertas et al., 1989] A. Huertas, W. Cole, and R. Nevatia. Using generic knowledge in analysis of aerial scenes: A case study. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1642-1648, Detroit, Michigan, August 1989.
- [Huertas et al., 1990] A. Huertas, W. Cole, and R. Nevatia. Detecting runways in complex airport scenes. *Computer Vision, Graphics, and Image Processing*, August 1990.
- [Jain, 1984] R. Jain. Segmentation of frame sequences obtained by a moving observer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:624-629, 1984.
- [Kim and Price, 1990] Y. C. Kim and K. Price. Multiple frame analysis of translation dominant motion. In *Proceedings of the DARPA Image Understanding Workshop*, Pittsburgh, Pennsylvania, September 1990.
- [Lin and Prasanna Kumar, 1990] W.-M. Lin and V.K. Prasanna Kumar. Parallel architectures for discrete relaxation algorithms. Technical report, University of Southern California, June 1990.
- [Lin, 1990] W.-M. Lin. Mapping image algorithms onto fixed size window architecture. USC Thesis in preparation, June 1990.
- [Lowe and Binford, 1982] D. Lowe and T. Binford. Segmentation and aggregation: An approach to figure-ground phenomena. In *Proceedings of the DARPA Image Understanding Workshop*, Stanford, California, September 1982.
- [Lowe and Binford, 1983] D. Lowe and T. Binford. The perceptual organization of visual images: Segmentation as a basis for recognition. In *American Association for Artificial Intelligence*, Washington, D.C., August 1983.
- [Lowe, 1985] D. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Press, 1985.
- [Medioni and Nevatia, 1984] G. Medioni and R. Nevatia. Matching images using linear features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):675-685, November 1984.
- [Medioni and Nevatia, 1985] G. Medioni and R. Nevatia. Segment-based stereo matching. *Computer Graphics and Image Processing*, 31(1):2-18, July 1985.
- [Menet et al., 1990] S. Menet, P. Saint-Marc, and G. Medioni. B-snakes: Implementation and application to stereo. In *Proceedings of the DARPA Image Understanding Workshop*, Pittsburgh, Pennsylvania, September 1990.
- [Mohan and Nevatia, 1988] R. Mohan and R. Nevatia. Perceptual grouping for the detection and description of structures in aerial images. In *Proceedings of the DARPA Image Understanding Workshop*, pages 512-526, Boston, Massachusetts, April 1988.
- [Mohan and Nevatia, 1989a] R. Mohan and R. Nevatia. Segmentation and description based on perceptual organization. In *Proceedings of the Con-*

- ference on Computer Vision and Pattern Recognition, pages 333-341, San Diego, California, June 1989.
- [Mohan and Nevatia, 1989b] R. Mohan and R. Nevatia. Using perceptual organization to extract 3-d structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1121-1139, November 1989.
- [Mohan, 1989] R. Mohan. *Perceptual Organization for Computer Vision*. PhD thesis, University of Southern California, August 1989. IRIS Technical Report 254.
- [Mundy et al., 1988] J. Mundy, A. Heller, and D. Thompson. The concept of an effective viewpoint. In *Proceedings of the DARPA Image Understanding Workshop*, pages 651-659, Cambridge, MA, 1988.
- [Nevatia and Babu, 1980] R. Nevatia and R. Babu. Linear feature extraction and description. *Computer Graphics and Image Processing*, 13:257-269, 1980.
- [Nevatia and Price, 1982] R. Nevatia and K. Price. Locating structures in aerial images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4(5):476-484, September 1982.
- [Ohlander et al., 1978] R. Ohlander, K. Price, and R. Reddy. Picture segmentation by a recursive region splitting method. *Computer Graphics and Image Processing*, 8:313-333, 1978.
- [Olsen, 1990] S. I. Olsen. Stereo correspondence by surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(3):309-315, March 1990.
- [Parvin and Medioni, 1989] B. Parvin and G. Medioni. A constraint satisfaction network for matching 3d objects. In *Proceedings of the International Conference on Neural Networks*, volume II, pages 281-286, Washington, D.C., June 1989.
- [Parvin and Medioni, 1990] B. Parvin and G. Medioni. A dynamic system for object description and correspondence. Submitted to International Conference on Computer Vision, 1990.
- [Peng and Medioni, 1988] S.-L. Peng and G. Medioni. Spatio-temporal analysis of velocity estimation of contours in an image sequence with occlusion. In *Proceedings of the International Conference on Pattern Recognition*, pages 236-241, Rome, Italy, November 1988.
- [Peng and Medioni, 1989] S.-L. Peng and G. Medioni. Interpretation of image sequences by spatio-temporal analysis. In *Proceedings of the Workshop on Visual Motion*, pages 236-241, Irvine, California, March 1989.
- [Ponce and Kriegman, 1989] J. Ponce and D. J. Kriegman. On recognizing and positioning curved 3-D objects from image contours. In *Proceedings of the DARPA Image Understanding Workshop*, pages 461-470, Palo Alto, CA, 1989.
- [Ponce, 1988] J. Ponce. Ribbons, symmetries, and skew symmetries. In *Proceedings of the DARPA Image Understanding Workshop*, pages 1074-1079, Cambridge, Massachusetts, 1988.
- [Rao et al., 1987] K. Rao, R. Nevatia, and G. Medioni. Issues in shape description and an approach for working with sparse data. In *Workshop on Spatial Reasoning and Multi-Sensor Fusion*, pages 168-177, Chicago, October 1987.
- [Rao, 1988] Kashipati Rao. *Shape Description from Sparse and Imperfect Data*. PhD thesis, University of Southern California, December 1988. IRIS Technical Report 250.
- [Reinhart and Nevatia, 1990] C. Reinhart and R. Nevatia. Efficient parallel processing in high level vision. In *Proceedings of the DARPA Image Understanding Workshop*, Pittsburgh, Pennsylvania, September 1990.
- [Rosenfeld et al., 1976] A. Rosenfeld, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6(6):420-453, June 1976.
- [Rosenfeld, 1986] A. Rosenfeld. Axial representations of shape. *Computer Vision, Graphics, and Image Processing*, 2(33):156-173, 1986.
- [Saint-Marc and Medioni, 1990] P. Saint-Marc and G. Medioni. B-spline contour representation and symmetry detection. In *First European Conference on Computer Vision*, pages 604-606, Antibes, France, April 1990.
- [Saint-Marc et al., 1989] P. Saint-Marc, J.-S. Chen, and G. Medioni. Adaptive smoothing: A general tool for early vision. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, San Diego, California, June 1989.
- [Stein and Medioni, 1990a] F. Stein and G. Medioni. Efficient two-dimensional object recognition. In *Proceedings of the International Conference on Pattern Recognition*, volume 1, pages 13-17, Atlantic City, New Jersey, June 1990.
- [Stein and Medioni, 1990b] F. Stein and G. Medioni. Toss - a system for efficient three dimensional object recognition. In *Proceedings of the DARPA Image Understanding Workshop*, Pittsburgh, Pennsylvania, September 1990.
- [Stevens and Brookes, 1987] K. Stevens and A. Brookes. Detecting structures by symbolic constructions on tokens. *Computer Vision, Graphics, and Image Processing*, 37(238-260), 1987.
- [Ulupinar and Nevatia, 1988] F. Ulupinar and R. Nevatia. Using symmetries for analysis of shape from contour. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 414-427, Tampa, Florida, December 1988.
- [Ulupinar and Nevatia, 1990a] F. Ulupinar and R. Nevatia. Inferring shape from contour for curved surfaces. In *Proceedings of the International Conference on Pattern Recognition*, volume 1, pages 147-154, Atlantic City, New Jersey, June 1990.
- [Ulupinar and Nevatia, 1990b] F. Ulupinar and R. Nevatia. Recovering shape from contour for shapes and edges.

In *Proceedings of the DARPA Image Understanding Workshop*, Pittsburgh, Pennsylvania, September 1990.

- [Vemuri and Aggarwal, 1986] B.C. Vemuri and J.K. Aggarwal. 3-D model construction from multiple views using range and intensity data. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 435-437, 1986.
- [Weiman and Chaikin, 1979] C. Weiman and G. Chaikin. Logarithmic spiral grids for image processing and display. *Computer Graphics and Image Processing*, 11(3):197-226, November 1979.
- [Witkin and Tenenbaum, 1983] A. Witkin and J. Tenenbaum. What is perceptual organization for? In *Proceedings of the International Joint Conference on Artificial Intelligence*, Karlsruhe, W. Germany, August 1983.
- [Wright and Ashford, 1989] P. Wright and N. Ashford. *Transportation Engineering: Planning and Design*. John Wiley and Sons, 1989.

IMAGE UNDERSTANDING RESEARCH AT GE

J. L. Mundy *

GE Corporate Research and Development Center
Schenectady, NY 12345

Abstract

Our research program in image understanding continues to emphasize the study of geometry and geometric relations for the representation and recognition of objects. A focus for the last year has been the development of a constraint-based geometric modeling system to represent generic object classes and to process constraints arising from camera viewpoint relationships.

Our work on model-based object recognition is also continuing with emphasis on test and evaluation of the recognition system in the context of aerial reconnaissance. In addition, some exciting new results have been obtained through the use of algebraic invariants which permit direct indexing of object models from scene features.

Object Recognition

Over the past several years, we have been developing a model-based vision system using a compact image feature, called the vertex-pair [Thompson and Mundy, 1987a]. The vertex-pair provides sufficient geometric information to determine the affine transformation between the image and scene coordinate frames with a computational cost proportional to N^2 , where N is the number of image vertices. Significant highlights of the development are:

- Initial demonstration of 3D object recognition from an unconstrained viewpoint, both indoor and outdoor scenes. - 1986
- Parallel implementation of the complete recognition algorithm on the Connection Machine, about 100X throughput increase [Thompson and Mundy, 1987b]. - 1987
- Development of automatic model vertex-pair selection based on geometric error performance as a function of viewpoint [Mundy *et al.*, 1988]. - 1988

- The recognition system is integrated with SRI's Cartographic Modeling Environment and Core Knowledge System. The introduction of viewing constraints and demonstration of the automatic generation of an airfield site database through object recognition [Corby *et al.*, 1988]. - 1989
- The collection of a series of airfield monitoring images and the execution of a benchmark test of recognition performance. Demonstration of recognition accuracy in excess of 98% [Heller and Mundy, 1990]. - 1990

The main focus of attention for the object recognition system in our current work is to extend the generality of image features which are used to determine model pose and to provide model hypothesis confirmation. The vertex-pair has proved to be quite effective under a wide variety of viewing conditions, but some valid object instances are incorrectly missed because not enough boundary vertices are recovered from the image data.

Another general area for improvement is the introduction of multi-resolution models. The current recognition system does not fail gracefully when image resolution is decreased. Our current algorithms for selecting model features does not account for the behavior of the image segmentation process. As the image resolution decreases, projected polyhedral features merge together and short edges disappear. Thus, it is necessary to account for both sensor properties and image segmentation behavior in generating model features. No single model feature set will suffice, and a series of models are needed which account for the entire space of camera viewpoints.

Another area which is being investigated in the area of object recognition is the introduction of general constraints on the relationship between objects and the relationship between objects and camera viewpoint. In our recent benchmark testing we took advantage of loose constraints on the camera viewpoint. For example, it is assumed that the camera is above the ground and within a range of elevation angle with respect to the ground surface normal; i.e., the camera viewpoint orientation is confined to a strip along latitudes on the view-sphere. It is now clear that a practical recognition system must be able to take advantage of a wide variety of site constraints which can improve recognition accuracy as well as considerably reduce the computation involved in searching the space of models and model pose param-

*Work at GE was supported in part by the DARPA Strategic Computing Vision Program in conjunction with the Army Engineer Topographic Laboratories under Contract No. DACA76-86-C-0007 and the Air Force Office of Scientific Research under Contract No. F49620-89-C-0033.

eters. The general area of constraints for recognition, as well as the development of constraint-based object models is emerging as a promising new area for object representation.

Constraint-Based Modeling

Another emphasis of our recent work is the representation of objects in terms of geometric constraints. The constraints are expressed symbolically in terms of relations between faces, edges and vertices of polyhedral models, as well as more general constraints between three dimensional position and orientation vectors. With a relatively small primitive set of constraint relations it is possible to construct any geometric relation on geometric entities. For example, the constraints associated with perspective viewing can be generated in terms of the collinearity relation between eyepoint, image point and world point. The collinearity relation can be defined by the more primitive operations associated with cross-product and dot-product on symbolic vector entities.

Our goal is to define a complete constraint representation and associated language so that any object model can be defined, including curved surface models and composite models involving generic constraint components. The representation is general enough to account for the constraints associated with multiple viewpoints and to allow the integration of image segmentation data derived from these such images. The general idea is that the constraint model represents a class of possible object configurations. A specific model instance is generated by minimizing a set of error metrics defined with respect to the empirical measurements taken from either manually selected image points or from automatically segmented image features. The final object instance is consistent with all of the specified model constraints and projected model features agree as well as possible with the actual image feature locations and orientations.

This general approach has evolved from our work in object recognition and the realization that the next generation recognition system must be able to accept generic models which can account for a large number of specific object configurations and can also represent a broad class of geometric relations which known to hold between objects and between objects and camera viewpoints.

Our work has progressed through a number of stages as summarized by the following items:

- Implementation and experimentation with a prototype system which used symbolic algebraic manipulation and a commercial nonlinear programming package from the IMSL library to solve the geometric constraint equations [Mundy *et al.*, 1989]. - 1988
- Investigation of a fully symbolic approach to the solution of constraint relations. Implementation of a prototype system based on the polynomial arithmetic routines provided by the GEOMETER system, which is being jointly developed by GE and UMass [Connolly *et al.*, 1989]. - 1989
- Implementation of a new constraint representation and new constraint solver which uses sym-

bolic methods to compile constraints into polynomials which are then solved by quadratic programming methods. Demonstration of the system on the construction of aircraft models from multiple orthographic views taken from mechanical drawings [Nguyen *et al.*, 1990]. - 1990

Our experience has indicated that robust convergence of the numerical methods can be achieved by careful attention to singularities and the use of pivoting to insure well conditioned solution matrices throughout the solution process. We have found that complex constraint systems can be solved with a small number of iterations and that it is not necessary to provide initial values for model parameters which are close to the final solution.

We have also observed that typical constraint systems lead to a block diagonal form for the Jacobian matrix. This result implies that the constraints can be partitioned into a set of loosely coupled groups. A major objective of our current work is to take advantage of this characteristic of geometric relations. This grouping will provide several advantages:

- Sparse matrix solution techniques can be invoked to considerably reduce the amount of computation required to solve the constraint system. It is estimated that the solution cost will go from N^3 down to approximately N by reducing the constraint Jacobian matrix to block diagonal form, where N is the number of constraint equations.
- The global error associated with empirical measurements can be reduced to a set of local error measures associated with each block. The error minimization process can then be directed at achieving error tolerance budgets within each group which will permit much tighter coupling between error minimization and model parameter adjustment. In effect, the error can "pushed" around the constraint network until worst-case tolerances are met.
- The partitioning of the constraints into a set of loosely coupled local networks is a prerequisite for mapping the constraint solution onto a parallel architecture. In view of the thousands of constraints which will be required to represent realistic scenes it will eventually be necessary to resort to parallel computation. Indeed, we have already explored the use of the CONVEX "mini-CRAY" architecture and found about a 10X improvement over a VAX-780 without resorting to explicit use of parallelism.

We have already demonstrated that general constraints can be readily applied to the representation of generic aircraft models where the required symmetries associated with the fuselage and the relation among airfoil surfaces and the fuselage can be expressed and solved by our constraint modeling system. At present however, it is somewhat tedious to specify constraints and considerable improvement is needed in the user interface. We intend to explore the use of general constraint components to reduce the modeling effort. It is also expected that a constraint programming language will prove to be a convenient approach to specifying most of the object representation.

It is also emphasized that nothing in the approach limits the representation to polyhedral models. Our initial emphasis on polyhedra is based on the fact that many practical applications can be effectively handled by polyhedral models. For example, we are applying our ideas in constraint-based modeling to the generation of models for image simulation with a focus on the application of mission rehearsal.

Object-Recognition With Algebraic Invariants

A joint project with the Robotics Research Laboratory at Oxford University on the use of algebraic invariants in model-based object recognition was initiated in the fall of 1988 [Forsyth *et al.*, 1990]. This new research area is especially important for progress in the use of constraint-based models for object recognition. The notion of algebraic invariants is that functions can be defined on groups of algebraic curves which are constant under viewpoint transformation. For example, invariants can be defined for two coplanar conics such that the conics can be characterized by a single number which can be derived from any projective transformation of the plane.

In order to insure that such invariants can be reliably derived from image data, we have developed a projectively invariant fitting procedure which insures that conics and other algebraic curves are derived according to the following requirement:

If an algebraic curve, C , is used to approximate a set of points, P , in the plane, then any projective transformation of the plane to a new set of points, P' , will yield a curve C' which is exactly the corresponding transformation of C .

The desired property follows by using an algebraic invariant to define the error measure for the data points. Thus any algorithm used to minimize the error with respect to the data points will produce a viewpoint invariant curve fit.

The use of invariants for coplanar conic curves has already been demonstrated through the implementation of a recognition system. We have shown that a library of more than twenty objects can be reliably recognized by indexing on algebraic invariant values. Our current emphasis is centered on the problem of deriving invariants for three dimensional configurations of curves and surfaces. There are a number of techniques for generating invariants which were developed during the last century which we are investigating. It also appears that a new approach may be feasible which is based on algebraic elimination theory.

Ultimately, it is planned to use invariants in the recognition of constraint-based models. It should be possible to derive invariants not only with respect to camera viewpoint but also invariants over the class of shapes which can be generated by a particular constraint model. The use of such invariants would seem necessary to reduce the dimension of the search space on model parameters. Current model-based recognition systems are hard pressed to determine the six parameters associated with model pose. Even simple constraint-models can have

twice this number and complex shapes may have hundreds of degrees of freedom. Clearly it will be necessary to derive invariant indices from image features to help reduce this search space.

References

- [Connolly *et al.*, 1989] Connolly, C.I., D. Kapur, J.L. Mundy, and R. Weiss, "GeoMeter: A System for Modeling and Algebraic Manipulation," *Proc. DARPA Image Understanding Workshop*, 1989, p. 797.
- [Corby *et al.*, 1988] Corby, N.R., J.L. Mundy, P.A. Vrobel, A.J. Hanson, L.H. Quam, G.B. Smith, and T.M. Strat, "PACE - An Environment for Intelligence Analysis," *Proc. DARPA Image Understanding Workshop*, 1988, p. 342.
- [Forsyth *et al.*, 1990] Forsyth, D.A., J.L. Mundy, A.P. Zisserman, and C.M. Brown, "Projectively invariant representations using implicit algebraic curves," *Proc. 1st European Conf. Comput. Vision*, Springer LNCS, 1990.
- [Heller and Mundy, 1990] Heller, A.J. and J.L. Mundy, "Benchmark Evaluation of a Model-Based Object Recognition System," *Proc. DARPA Image Understanding Workshop*, 1990.
- [Mundy *et al.*, 1988] Mundy, J.L., A.J. Heller, D.W. Thompson, "The Concept of an Effective Viewpoint," *Proc. DARPA Image Understanding Workshop*, 1988, p. 651.
- [Mundy *et al.*, 1989] Mundy, J.L., P.A. Vrobel, and R.E. Joynson, "Constraint-Based Modeling," *Proc. DARPA Image Understanding Workshop*, 1989, p. 425.
- [Nguyen *et al.*, 1990] Nguyen, V., J.L. Mundy, and D. Kapur, "Modeling Polyhedra by Constraints," *Proc. DARPA Image Understanding Workshop*, 1990.
- [Thompson and Mundy, 1987a] Thompson, D.W. and J.L. Mundy, "Three-Dimensional Model Matching From an Unconstrained Viewpoint," *Proc. IEEE Robotics and Automation*, 1987, pp. 280.
- [Thompson and Mundy, 1987b] Thompson, D.W. and J.L. Mundy, "Model-Directed Object Recognition on the Connection Machine," *Proc. DARPA Image Understanding Workshop*, 1987.

IMAGE UNDERSTANDING RESEARCH AT HONEYWELL

Bir Bhanu

Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

ABSTRACT

Image understanding research at Honeywell is directed towards autonomous/semiautonomous vehicle navigation, automatic target recognition, and image exploitation. The objective of our work is to develop robust and efficient image understanding techniques for these applications operating from various platforms. The topics currently under investigation are: machine learning for multi-level vision system; inertial navigation sensor integrated motion, binocular stereo, and laser radar techniques for obstacle detection; qualitative 3-D scene modeling for dynamic scene understanding, motion analysis, and target tracking; map-based automatic target recognition and tracking; and multisensor target recognition. This paper summarizes the progress made in some of these key areas during the period from March 1989 to June 1990. We also briefly discuss the important topic of image database and scientific performance evaluation of vision algorithms and systems.

1. INTRODUCTION

This paper provides an overview of the research performed by our group during the past year. Our research in image understanding is directed towards model-based target recognition and machine learning and knowledge-based interpretation of scene dynamics. The key accomplishments achieved during the past year are as follows. We have developed an adaptive image segmentation technique that has shown improvements of 30-50% over state-of-the-art nonadaptive segmentation approaches. We have demonstrated the concepts for a multistrategy machine learning system for target recognition, target model acquisition, and refinement. We have also shown that robust range estimates can be obtained by inertial navigation sensor integrated motion analysis. In addition, we have performed experiments for binocular and motion stereo integrated system for dense ranging.

We have investigated the following major topics:

- (1) *Machine learning* for multi-level vision system for adaptive segmentation, target recognition, target model acquisition, and target model refinement.
- (2) *Inertial navigation sensor integrated* motion, binocular stereo, and laser radar techniques for obstacle detection.
- (3) *Qualitative* scene understanding for dynamic scene and motion analysis for target motion detection and tracking, *dynamic model matching* for landmark recognition, and *hierarchical symbolic grouping* for interpretation of terrain.

The synopsis of the technical achievements and key ideas in each of these areas is given below. We also present

a brief discussion on image database and scientific performance evaluation of vision algorithms and systems.

2. MACHINE LEARNING FOR MULTI-LEVEL VISION

Present target recognition systems are unable to adapt to changes in environmental conditions, target variations, and the unexpected appearance of new targets. Each of these situations affects the appearance of the targets in the image, which in turn, degrades the overall performance of current generation recognition systems.

One of the key challenges to automating the target recognition process is that of automatically responding to changes occurring in the targets seen in an image. We address this problem at every stage of the multi-level vision problem by a unique multi-strategy machine learning approach not available in any current model-based recognition system. We want to show that significant benefits can accrue by applying machine learning technology to automatically recognize targets, acquire new target models and update their descriptions, learn new target features based on perceptual cues, and to adapt segmentation parameters based on changing environmental conditions.

Through an in-depth analysis performed by Honeywell¹ on the applicability of state-of-the-art machine learning technology to model-based vision, we have developed the concepts for a novel machine learning system called TRIPLE (*Target Recognition Incorporating Positive Learning Expertise*). At the high level of computer vision, TRIPLE uses *explanation-based learning (EBL)* and *structured conceptual clustering (SCC)* in the target recognition and learning process. During the intermediate level vision processing, TRIPLE uses *explanation-based learning* with a perceptual cue database to acquire new target features. Finally, at the low level of vision, TRIPLE uses *genetic algorithms* for parameter adaptation capability. Thus, the TRIPLE system provides a learning capability at all three levels of vision: low, intermediate, and high.

Addition of the machine learning techniques listed above to each level of the computer vision process yields the target recognition system shown in Fig. 1. Each stage of the vision process now contains a localized learning loop that provides the adaptive behavior necessary at each level. Further, each level of the process is able to interact with the levels immediately above and below it. This communication allows each level to obtain extra or missing information from the previous level and also allows feedback on data quality to be passed from one level down to the next. For example, at the high level of vision, the classification process may request additional feature information from the intermediate level in order to correctly recognize a target. The request

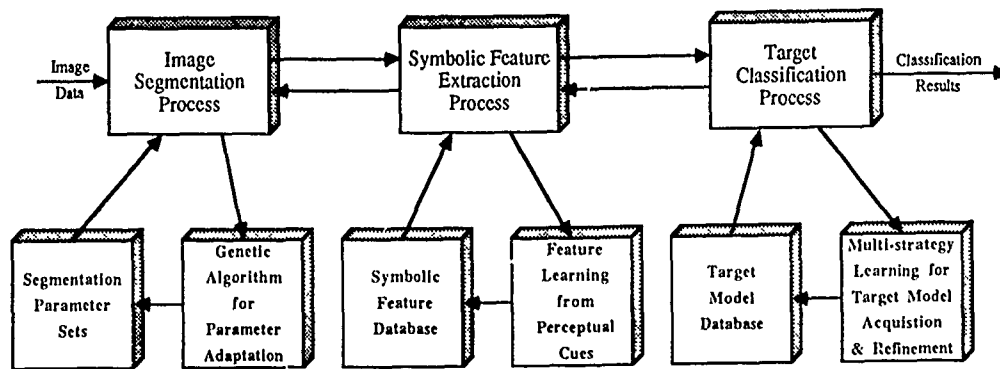


Fig. 1: Target recognition system incorporating machine learning techniques at each level of processing to enhance recognition performance.

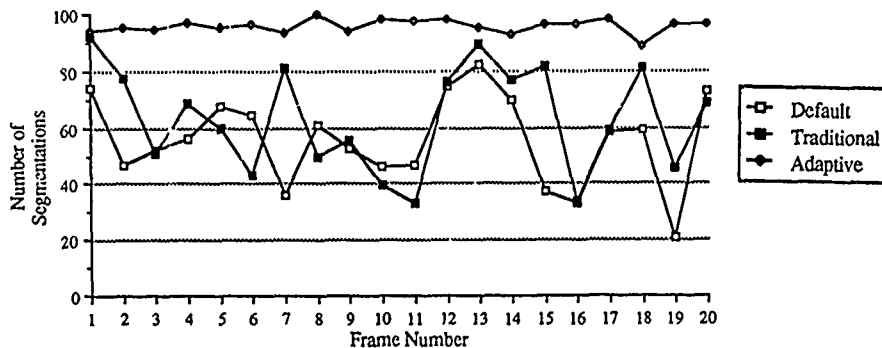


Fig. 2: Comparison of adaptive image segmentation system with default Phoenix²⁴ parameters and the traditional segmentation approach commonly used in the Computer Vision field.

may, in turn, require the intermediate level to obtain further region and edge information from the low level. After the classification results have been computed, the high level process evaluates the usefulness of the features that were used during the classification and sends this information to the intermediate level. Finally, the intermediate level must also supply feedback data to the low level process so that the utility of the segmentation parameters can be determined.

Some of the other applications of machine learning that we are working on include intent recognition and automated knowledge acquisition for image exploitation.

2.1 Adaptive Image Segmentation

Image segmentation is typically the first, and most difficult, task of any automated image understanding process. All subsequent interpretation tasks, including feature extraction, target detection, and target recognition, rely heavily on the quality of the segmentation process. Despite the large number of segmentation techniques presently available, no general methods have been found which perform adequately across a diverse set of imagery. Only after numerous modifications to an algorithm's control parameter set can any current method be used to process the wide diversity of images encountered in dynamic outdoor applications such as the operation of an autonomous robotic land/air vehicle, automatic target recognizer, or a photointerpretation task.

The image segmentation problem is characterized by several factors which make the parameter selection process very difficult. These factors include numerous control parameters, lack of precise segmentation algorithm models,

and problems associated with the evaluation of segmentation.

We have developed an adaptive image segmentation system for the segmentation of color images based on a genetic algorithm which can adapt to changes in the environment (time of day and weather conditions). The genetic algorithm efficiently searches the enormous hyperspace of segmentation parameter combinations using a collection of search points known as a population. By combining high performance members of the current population to produce better parameter combinations, the genetic algorithm is able to locate the parameter set which maximizes the segmentation quality criteria. Fig. 2 presents the comparison of three approaches on a database of 20 images. The *default parameters* have been suggested after extensive amounts of testing by various researchers who developed the Phoenix algorithm.²⁴ The parameters for the *traditional approach* are obtained by manually optimizing the segmentation algorithm on the first image in the database and then utilizing that parameter set for the remainder of the experiments. This approach to segmentation quality optimization is currently standard practice in state-of-the-art computer vision systems. The average segmentation quality for the adaptive segmentation technique was 95.8% (average of 100 experiments). In contrast, the performance of the default parameters was only 55.6% while the traditional approach provided 63.2% accuracy. As the figure shows, the performance of both of these alternative approaches was highly erratic throughout the sequence.

This is the first segmentation approach to incorporate adaptation in a closed-loop feedback system and will have

significant benefits for signal processing and adaptive control applications requiring robust performance in unstructured environments. The basic adaptation methodology is applicable to imagery from other sensors and can utilize any segmentation algorithm. The papers by Bhanu, Lee, and Ming⁷⁻¹⁰ provide details of the adaptive image segmentation process.

2.2 Target Recognition, Target Model Acquisition and Refinement

A major technology gap in state-of-the-art model-based target recognition for outdoor scenes is the process of model (natural or man-made) acquisition. Generally target models are fixed and recognition systems do not have any learning capability; therefore, they are not adequate by themselves for target recognition in dynamic environments.

Due to recent advances in machine learning technology, some of the problems encountered in the target recognition domain seem to be resolvable. Learning allows an intelligent recognition system to use situation context in order to understand images. This context, as defined in a machine learning scenario, consists of a collected body of background knowledge as well as environmental observations which may impact the processing of the scene.

Machine learning facilitates two main advances in the target recognition domain: automatic knowledge base acquisition and continuous knowledge base refinement. The use of learning during knowledge base construction will save the user from spending the large amount of time necessary to derive target models and databases. Knowledge base refinement can then be incorporated to make any necessary changes to improve the performance of the recognition system. These two modifications alone will serve to significantly advance the present abilities of most target recognition applications.

At high level of vision, TRIPLE combines EBL and SCC learning methodologies with a knowledge-based matching technique to provide robust target recognition. Explanation-based learning provides the ability to build a generalized description of a target class using only one example of that class. Structured conceptual clustering allows the recognition system to classify a target based on simple, conceptual descriptions rather than using a predetermined, numerical measure of similarity. While neither method, used separately, would provide substantial benefits to a target recognition system, they can be combined to offer a consolidated technique which employs the best features of each method and is very robust.

Using dynamic models, TRIPLE can recognize targets present in the database. If necessary, the models can be refined if errors are found in the representation. Additionally, TRIPLE can automatically store a new target model and recall it when that target is encountered again. Finally, since TRIPLE uses qualitative data structures to represent targets, it can overcome problems such as image noise and occlusion. The papers by Bhanu and Ming^{11,25} provide more details of the TRIPLE system for target model recognition, acquisition, and refinement.

3. INS INTEGRATED TECHNIQUES FOR OBSTACLE DETECTION

We are working on both passive and active (laser radar) techniques for obstacle detection for ground vehicles and helicopters. In the following, we describe two passive ranging techniques.

3.1 INS Integrated Motion Analysis

Land navigation requires a vehicle to steer clear of trees, rocks, and man-made obstacles in the vehicle's path while vehicles in flight, such as rotorcraft, must avoid antennas, towers, poles, fences, tree branches, and wires strung across the flight path. Automatic detection and recognition of these obstacles by passive/active sensors and the necessary guidance and control actions triggered by such detection would facilitate autonomous vehicle navigation.

Many types of existing vehicles contain inertial navigation systems (INS) which can be utilized to greatly improve the performance of several computer vision applications such as obstacle detection, target motion detection, target tracking, binocular stereo, etc. and make them useful for practical military and civilian applications. As an example, motion analysis techniques can effectively use the output of an INS to improve interest point selection, matching of the interest points, and the subsequent motion detection, tracking, and obstacle detection.

We are using INS measurements to enhance the quality and robustness of motion analysis techniques. The key ideas of our maximally passive approach for obstacle detection¹² are the use of constraints from INS information for improved correspondence, texture-based scene analysis for the selection of uniformly distributed interest points, the concepts of match and range confidences, smoothing of range values over multiple frames, and the selective application of a laser sensor. Details of the INS integrated motion analysis for obstacle detection are given in the paper and reports by Bhanu, Roberts, and Ming.^{13-15,29}

3.2 Binocular and Motion Stereo for Dense Ranging

Range measurements to objects in the world, relative to mobile platforms such as ground or air vehicles, is critical for visually aided navigation and obstacle detection/avoidance. Active (laser) range sensors can be used to provide such range measurements although they have a limited field of view, suffer from slow data acquisition, and are expensive. We address the development of a robust and efficient *passive* technique for obtaining range measurements. Our approach consists of a synergistic combination of two types of passive ranging: binocular stereo and motion stereo.³⁰ The problem that we address is the optimal combination of sparse motion stereo range estimates, $r_m(x,y)$, and sparse binocular stereo range estimates, $r_b(x,y)$, so the resulting range map is as accurate and dense as possible throughout the entire field of view.

Binocular stereo and motion stereo compute range to "distinguished" points in the image. Binocular stereo range computations suffer the greatest error at the edges of the camera's field of view (FOV), where motion stereo range is most accurate. The converse error relationship holds true in the vicinity of the focus of expansion where motion stereo range error is great and binocular stereo range error is small.

We have developed detailed error models for binocular and motion stereo and inertial reference unit, and developed a Kalman and Blending filter. The Kalman filter's binocular stereo measurement consists of motion stereo range subtracted from binocular stereo range. The filter's motion stereo measurement is the negative of the filter's binocular stereo measurement. The coincident points of interest, i.e. those points for which range is computed by both motion and binocular stereo techniques, are used as measurements to estimate errors in the ranging processes. The points in the range maps which are not coincident can be corrected with these error estimates, improving the overall quality of the

composite range map. The initial results of our work are reported in the paper by Symosek et al.³⁰

4. ROBOTIC VEHICLE NAVIGATION

4.1 Qualitative Motion Detection and Tracking

We have developed a unique approach called DRIVE (Dynamic Reasoning from Integrated Visual Evidence) based on *qualitative reasoning and modeling* for target motion detection and tracking.^{3,6,19-22} The DRIVE system performs dynamic scene understanding needed to support the application of smart weapons and autonomous navigation of robotic vehicles. Instead of refining a single quantitative description of the observed environment over time, multiple qualitative interpretations of the scene are maintained simultaneously. This technique offers considerable flexibility over traditional numerical techniques which are often ill-conditioned and noise-sensitive. With DRIVE, an autonomous system can (i) detect and classify moving targets in the scene, (ii) estimate the vehicle's egomotion, and (iii) derive the 3D structure of the stationary environment.

The 3-D motion of targets is obtained from (a) displacement vectors of point features without any knowledge about the underlying 3-D structure, (b) discovering inconsistencies between the current state of the qualitative 3-D scene model and the changes actually observed in the scene, and (c) by detecting moving edges and regions.^{4,18}

DRIVE uses a new algorithm for computing the region of possible focus-of-expansion (FOE) locations in image sequences, called the fuzzy FOE.^{21,23} This computation is accomplished in a unique manner by separating the rotational and translational components of the vehicle's motion and using a robust method for computing the displacement vector between two images using adaptive windows.¹⁸ The technique is applicable to platforms with no on-board INS.

The 'fuzzy' FOE algorithm allows the direction of instantaneous heading of an autonomous land vehicle to be precisely determined within 1° using image information exclusively. The results obtained using ALV imagery taken at five different sites demonstrate the algorithm's performance capabilities. This result has significant scientific importance for targeting applications. It allows the determination of self motion of moving imaging devices. Rotation in the horizontal and vertical directions (pan and tilt only) of $\pm 5^\circ$ or larger can be successfully handled by the algorithm.⁵ Moreover, it allows the use of *passive approaches* for surveillance activities that must detect and track moving targets and must detect and avoid obstacles using passive sensors mounted on a mobile platform.

We have developed preliminary algorithms to integrate the DRIVE system with digital terrain elevation and land cover data. These algorithms provide information about the map location of the moving targets, the road label on which the targets are possibly traveling, and neighboring landmarks. Such information is desired for military applications and we have performed initial experiments to establish its usefulness in detecting moving targets in both high clutter and low contrast situations. The paper by Bhanu et al.¹⁸ provides details of the interest point selection, disparity analysis, fuzzy FOE, qualitative scene model, map-based tracking, and edge/region based approach.

4.2 Dynamic Model Matching for Landmark Recognition

We have developed a technique called PREACTE (Perception REASONing ACTION and Expectation) based on *dynamic model matching* for landmark recognition from a mobile platform.²⁶⁻²⁸ The technique can recognize landmarks

and other targets from partial and complete views in dynamic scenarios. It relies on the generation of multiple landmark descriptions from 3D models based on different estimated ranges and aspect angles. These descriptions are a result of feature, spatial, and geometric models of a single landmark. Expectations about the landmarks (appearances) vary dynamically as the autonomous robot approaches the landmark. Dynamic Model Matching also includes the generation of specific landmark recognition planning strategies whereby different features of different landmarks are emphasized at varying ranges. It is an expectation driven, knowledge-based approach and uses limited map information for updating the robotic vehicle's location in the map.

4.3 Interpretation of Terrain

An autonomous robotic vehicle must be able to navigate not only on the roads, but also through terrain in order to execute its missions of surveillance, search and rescue, and munitions deployment. To do this, the vehicle must categorize the terrain regions it encounters as to the trafficability of the regions, the land cover of the regions, and region-to-map correspondence. Our approach for terrain interpretation employs a robust texture-based algorithm and a hierarchical region labeling scheme for ERIM 12 channel Multi-Spectral Scanner data. The technique, called HSGM (Hierarchical Symbolic Grouping for Multi-spectral data), is specifically designed for multi-spectral imagery, but is appropriate for other categories of imagery as well. For this approach, features used for segmentation vary from macro-scale features at the first level of the hierarchy to micro-scale features at the lowest level. Examples of labels at the macro-level are sky, forest, field, mountain, road, etc.

For each succeeding level of the hierarchy, the identified regions from the previous stage are further subdivided, if appropriate, and each region's labeling is made more precise.

The process continues until the last stage is reached and no further subdivision of regions from the preceding stage appears to be necessary. Examples of region labels for this level of the hierarchy are gravel road, snowberry shrub, gambel oak tree, rocky ledge, etc. Further development of the technique will employ multiple sources of *a priori* information such as land cover, terrain elevation map information, range data, seasonal information, and time of day. Details of the HSGM technique with results and examples from real imagery are given in papers by Bhanu and Symosek.^{16,17}

4.4 Vision-based Targeting Experiments

As discussed earlier, we have developed two key algorithm suites, called DRIVE (Dynamic Reasoning from Integrated Visual Evidence) and PREACTE (Perception, REASONing, ACTION and Expectation). DRIVE accomplishes target motion detection and tracking while PREACTE performs landmark recognition. We plan to advance this research by performing a set of scientific experiments directed towards a practical mobility and targeting application of a robotic combat vehicle.

We plan to conduct scientific experiments in two areas: landmark recognition for path traversal and target motion detection and tracking. Two series of experiments are planned, one in each of these areas. Each experimental series begins with data collection and proceeds through progressively more difficult scenarios. The final experiments in the series will be characteristic of practical mobility scenarios for a robotic combat vehicle. For both series of experiments, the vehicle will be in continuous motion.

Landmark recognition experiments include laboratory

landmark recognition tests using off road data; non-real time landmark recognition in off road traversal by the robotic vehicle; real time dynamic landmark recognition in off road traversal by the robotic vehicle; and dynamic landmark model learning with return path traversal. Motion detection and tracking experiments involve verifying motion results against land navigation data; non-real time detection of multiple moving targets while maintaining reasonable rotation components of the vehicle; real time detection of multiple moving targets; integrating ETL map data with target motion detection and tracking; and advanced experiments carried out under more difficult visual scenes involving low contrast and high clutter.

We also plan to develop a flexible software architecture and the associated software for "real time" instrumentation and evaluation of the landmark recognition and the motion detection and tracking algorithms. Some of the important aspects of this work involve defining the criteria for evaluation and acquiring, retrieving, and presenting the desired information in meaningful ways so as to provide insight into the associated vision algorithms.

5. IMAGE DATABASE AND SCIENTIFIC PERFORMANCE EVALUATION

At present, very little work has been done in the area of performance evaluation for image understanding algorithms and systems. In the DARPA-sponsored image understanding research, a wide variety of algorithms and systems are being developed for photointerpretation, navigation, manufacturing, cartography, and targeting applications. Quantitative and qualitative scientific performance evaluation methods for vision algorithms and systems will provide an effective way to scientifically measure the progress being made by the computer vision field. The development of an effective evaluation methodology will lead to more rapid technology transfer to DoD applications by providing the means to assess readiness of the technology. In addition, the technology development timeline will shrink once a means exists to clearly evaluate the performance of vision algorithms and systems. All this will help in advancing computer vision field at a faster pace.

The critical ingredients for scientific performance evaluation are:

- (a) Image database and associated groundtruth information,
- (b) Techniques for performance evaluation,
- (c) Common system environments (KBVision and others).

The primary objectives of the Image Database/Performance Evaluation are to establish a national research database of computer vision imagery and develop a performance evaluation capability for "matured" IU algorithms. The database will be accessible to all members of DARPA's IU community over the Arpanet through a set of uniform access procedures. A taxonomy of computer vision research will be generated to characterize computer vision algorithms and systems for the purpose of database indexing and evaluation. A set of techniques and models for algorithm/system performance evaluation of selected "matured" algorithms will be developed to facilitate the uniform comparison of algorithms. Performance evaluation provides performance analysis (strengths/weaknesses), sensitivity analysis, and performance models. All these lead to prediction of performance of algorithms and prediction is an important element of science.

Through active interaction with the DARPA IU/SC community, the following objectives are pursued for scientific

performance evaluation.

- (a) Creation of a standardized image database to be used for performance evaluation of diverse vision algorithms and systems. Generation of a taxonomy for vision research based on applications, scientific principles, functionality, and software/hardware variations. Design and development of an extensible database system accessible over the Arpanet.

Establishment of a scientific methodology for performance evaluation of model-based vision systems. Definition of a standard terminology and establish benchmarks for performance evaluation of algorithms and systems. Implementation and evaluation of the baseline model-based vision system.

5.1 National Research Database of Computer Vision Imagery

The objective of establishing a national research image database is to promote the orderly development and dissemination of image information to serve the needs of DARPA IU algorithms/systems developers. This encompasses the standards for data interchange and activities for data collection, data organization, and design and development of an extensible database system accessible to IU researchers over the Arpanet.

The important considerations for these databases are: ground truth data requirements (site, sensor characterization, sensor platform, targets of interest, meteorological conditions), ground truth recording procedures, and database quantity/quality/variety requirements. The ground truth information is very critical and many times is not available or is too expensive to capture. Whenever the ground truth information is available, imagery should be partitioned into two categories: For some imagery, the ground truth is supplied to the researcher so that he/she can use them in the development of vision algorithms; the other category should be the imagery for which ground truth is sequestered and used to evaluate the robustness of the algorithms after development.

One potential use of an accepted imagery database would be for evaluating various "matured" algorithms that perform the same function (e.g., stereo, segmentation, motion detection, target recognition in range images, etc).

A current detailed taxonomy of vision research based on diverse criteria is desired.² The rationale for characterization is to help in the organization and development of image database, definition of benchmarks, and methodologies for evaluation. This characterization will provide a common framework of terminology and description to promote improved communication among the members of the vision community and between technology developers and appliers. Since the computer vision field is still quite young and undergoing rapid evolution, the proposed taxonomy should be viewed as a "snapshot" of the field today and will likely need to undergo significant modifications and extensions as the field progresses. After the development of the proposed taxonomy, the development of the other goals will be pursued: a common image database, general vision system benchmarks, and an effective methodology for performance evaluation. One can think of a very deep tree whose leaf nodes are very specific (for example, the segmentation of tank targets at "close" distances in range images for terminal homing applications) We associate the specific database, benchmarks, and methodology with these leaf nodes for performance evaluation.

5.2 Scientific Methodology and Models for Performance Evaluation

Since one of the goals of computer vision is to build machines that can solve real world problems, we need to define the systematic methods and models for performance evaluation of individual vision algorithms (segmentation, feature computation, texture measurement, etc.) and systems (target recognition, vision-based navigation, etc.) for a particular application (terminal homing, surveillance, etc.).²

It is important to have common terminology and benchmarks for performance evaluation. Subtle differences in meaning can be very important for evaluation. A lexicon that establishes standard terminology and standard benchmarks will provide uniformity in carrying out scientific experiments for performance evaluation. The emphasis of performance evaluation is on computer vision problems, scientific *experimental design* and *interfaces* between vision components and functions. We need to define a performance metric for each of the image understanding algorithms as well as a performance metric for the system as a whole. This will be done for the specific matured algorithms/systems (model-based vision for target recognition) being pursued by the Image Understanding community.

REFERENCES

- B. Bhanu, "Machine Learning in Computer Vision," Technical Report, Honeywell Systems and Research Center (1988).
- B. Bhanu, "Understanding Scene Dynamics," *Proc. DARPA Image Understanding Workshop*, pp. 147-164 (May 1989).
- B. Bhanu and W. Burger, "Qualitative Motion Detection and Tracking of Targets from a Mobile Platform," *Proc. DARPA Image Understanding Workshop*, pp. 289-318 Morgan Kaufmann Publishers, (April 1988).
- B. Bhanu and W. Burger, "A System for Motion Detection and Tracking of Targets from a Mobile Platform," *Patent granted*, (1990).
- B. Bhanu and W. Burger, "A System for Computing the Self Motion of Moving Imaging Devices," *Patent granted*, (1990).
- B. Bhanu and W. Burger, "Qualitative Approach for Dynamic Scene Understanding," *Computer Vision, Graphics and Image Processing (Accepted)*, (To Appear 1990).
- B. Bhanu, S. Lee, and J. Ming, "Adaptive Image Segmentation Using A Genetic Algorithm," *Proc. DARPA Image Understanding Workshop*, pp. 1043-1055 Morgan Kaufmann Publishers, (May 1989).
- B. Bhanu, S. Lee, and J. Ming, "A System for Adaptive Image Segmentation," *Patent Pending*, (1989).
- B. Bhanu, S. Lee, and J. Ming, "Self-Optimizing Control System for Adaptive Image Segmentation," *Proc. DARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, (September 1990).
- B. Bhanu, S. Lee, and J. Ming, "Adaptive Image Segmentation," *Submitted to IEEE Trans. on Pattern Analysis and Machine Intelligence*, (February 1990).
- B. Bhanu and J.C. Ming, "TRIPLE: A Multi-Strategy Machine Learning Approach to Target Recognition," *Proc. DARPA Image Understanding Workshop*, pp. 537-547 (April 1988).
- B. Bhanu and B. Roberts, "Inertial Navigation Sensor Integrated Obstacle Detection System," *Patent Pending*, (1989).
- B. Bhanu and B. Roberts, "Obstacle Detection During Rotorcraft Low Altitude Flight and Landing," Second Annual Technical Report for NASA-Ames (July 1990).
- B. Bhanu, B. Roberts, and J. Ming, "Inertial Navigation Sensor Integrated Motion Analysis," *Proc. DARPA Image Understanding Workshop*, pp. 747-763 Morgan Kaufmann Publishers, (May 1989).
- B. Bhanu, B. Roberts, and J. Ming, "Automatic Obstacle Detection and Avoidance by Helicopters," *Proc. 1990 IEEE International Conference on Robotics & Automation*, pp. 954-959 (May 1990).
- B. Bhanu and P. Symosek, "Interpretation of Terrain Using Hierarchical Symbolic Grouping for Multi-Spectral Images," *Proc. DARPA Image Understanding Workshop*, pp. 466-474 (Feb. 1987).
- B. Bhanu and P. Symosek, "Interpretation of Terrain Using Multispectral Images," *Submitted to Pattern Recognition*, (1989).
- B. Bhanu, P. Symosek, J. Ming, W. Burger, H. Nasr, and J. Kim, "Qualitative Target Motion Detection and Tracking," *Proc. DARPA Image Understanding Workshop*, pp. 370-398 Morgan Kaufmann Publishers, (May 1989).
- W. Burger and B. Bhanu, "Qualitative Motion Understanding," *Proc. Tenth International Joint Conference on Artificial Intelligence, IJCAI-87, Milan, Italy*, Morgan Kaufmann Publishers, (August 1987).
- W. Burger and B. Bhanu, "Dynamic Scene Understanding for Autonomous Mobile Robots," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 736-741 (June 1988).
- W. Burger and B. Bhanu, "On Computing a 'Fuzzy' Focus of Expansion for Autonomous Navigation," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 563-568 (June 1989).
- W. Burger and B. Bhanu, "Qualitative Understanding of Scene Dynamics for Autonomous Mobile Robots," *The International Journal of Robotics Research* 9(6)(December 1990).
- W. Burger and B. Bhanu, "Estimating 3-D Egomotion from Perspective Image Sequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence (Accepted)*, (To Appear 1990).
- K.I. Laws, "The Phoenix Image Segmentation System: Description and Evaluation," SRI International Technical Note No. 289 (December, 1982).
- J. Ming and B. Bhanu, "A Multistrategy Learning Approach for Target Model Recognition, Acquisition and Refinement," *Proc. DARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, (September 1990).
- H. Nasr and B. Bhanu, "Dynamic Model Matching for Target Recognition from a Mobile Platform," *Proc. DARPA Image Understanding Workshop*, pp. 527-536 (April 1988).
- H. Nasr and B. Bhanu, "Landmark Recognition for Autonomous Mobile Robots," *Proc. IEEE International Conference on Robotics and Automation*, pp. 1218-1223 (April 1988).
- H. Nasr and B. Bhanu, "Landmark Recognition System Using Dynamic Model Matching," *Patent pending*, (1988).
- B. Roberts and B. Bhanu, "Inertial Navigation Sensor Integrated Motion Analysis for Autonomous Vehicle Navigation," *Proc. DARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, (September 1990).
- P. Symosek, B. Bhanu, S. Snyder, and B. Roberts, "Motion and Binocular Stereo Integrated System for Passive Ranging," *Proc. DARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, (September 1990).

Image Understanding Research at Rochester

Christopher Brown, Randal Nelson
Computer Science Department
University of Rochester
Rochester, NY 14607

Abstract

Animate (real time, purposeful, active) vision requires parallel computing of several varieties (MIMD, SIMD, pipelined). This year's research has been devoted, on the applications side, to sophisticated real time vision algorithms in the areas of gaze control, optic flow analysis, colored object recognition and location, and selective attention. This year the parallel computing environment has been able to support users on the Psyche and Platinum NUMA operating systems, the Zebra and Zed pipeline parallel programming tools, and the Instant-replay and Moviola debugging and performance monitoring toolkits. Goals for next year include integrating cognitive planning with dextrous manipulation and real time vision, and incorporating increasingly sophisticated multi-agent interaction in animate vision systems.

1 Vision Applications

Vision applications are an important part of our research in parallel, animate, real-time approaches to vision. The laboratory is expanding with the addition of a Utah 16-dof dextrous manipulator, and we are designing a system to integrate the full spectrum of AI techniques, from cognitive planning down to real-time control. This year we have concentrated on sophisticated vision algorithms that use state of the art hardware, with the aim of achieving responses fast enough to interact with the world. Paul Chou's Highest Confidence First (HCF) algorithm for Markov Random Fields was extended to the parallel Local HCF algorithm, which is ideally suited to fast implementation on the Connection Machine. Relaxation algorithms for general arc consistency were implemented on the CM by Cooper and Swain at Syracuse's DARPA-funded NPAC, as was an optimized version for high level structure recognition. Pipelined parallel computation is used heavily in Nelson's movement detector (which detects objects moving with respect to a possibly moving background) and Swain's colored object recognizer (see below).

Several parallel vision applications were pursued this year, including Butterfly programming, Markov Random

Field and connectionist applications, and work aimed at integrating the real-time laboratory and using it for complex planning tasks that include sensing and acting. Key papers are [Simard 1990 (Recurrent backpropagation); Brown et al. 1989 (decentralized Kalman filters); Aloimonos and Brown 1989 (Robust computation of intrinsic images); Chou and Brown 1990 (Sensor fusion, reconstruction and labeling); Wixson and Ballard 1990 (Color histograms); Rimey and Brown 1990 (Hidden Markov models); Yamauchi 1989 (Juggler); Nelson 1990 (Movement detection); Brown 1990a (predictive control); Swain 1990 (Thesis on colored object recognition and location); Martin et al. 1990 (ARMTRAK project).

1.1 Real-time Colored Object Search and Recognition

Recent work on fast object detection uses color histogram matching techniques and relational modeling. Almost all current object recognition schemes require that image features be matched to model features, requiring a time polynomial in the number of features to perform the matching. By adding an initial stage that does not perform pose calculation but rather simply detects the likely presence of the object in the image, considerable efficiency can be gained. The idea is that this initial stage would be used to rank each gaze in a set of candidate gazes according to the likelihood that the image produced by the gaze contains the desired object. This ranking can then be used to choose the order in which a more sophisticated object recognition program (which would calculate pose) should be applied to the candidate images.

Wixson [Wixson and Ballard 1990] and Swain [Swain and Ballard, 1989] use object detection schemes that rely on the assumption that the color histogram of an object can be used as an object "signature" that is invariant over a wide range of scenes and object poses. The color histogram is computed by the Datacube parallel hardware, and an efficient matching algorithm due to Swain can locate the best match in data bases containing between 10 and 70 items in a constant 15 milliseconds.

Wixson's object search algorithm gazes around the room and matches what it sees with the database, making a spatial map of object locations, and can also adjust its gaze position to improve the goodness of the match

(a good example of active vision). Wixson is also using knowledge of expected spatial relationships to direct the search process. Indirect search uses a finite set of relationships (FRONT-OF, NEAR, LEFT-OF, etc.) between objects. The relationships may be known apriori or, more interestingly, derived from experience with the scene. Characterizing the occurrence of relationships as Bernoulli trials leads to a confidence interval representation of the probability of the relations holding. In turn, these probabilities can be used in a "highest impact first" search that acquires information in the order that maximally decreases expected uncertainty. The result is to derive Garvey-like strategies on the fly from first principles, while supporting learning.

In his thesis Swain investigates color cues for object recognition. He has developed a robust matching algorithm called histogram intersection that can recognize nonrigid, occluded colored objects from a large database (up to 70 items so far, but that is not an upper bound). It runs in time linear in the number of database entries, taking 38 ms for a 19-object database and 150 ms. for 70 objects. An incremental version of histogram intersection relies on the fact that most histogram bins are unnecessary in a match, and can perform recognition in a constant 15 ms for databases at least up to 70 objects in size. In one typical run, 30 of 32 objects were correctly identified from a catalog of 66 objects, while for the remaining two objects the second best match was correct. The instance views have different viewing angles from those that generated the catalog. The demonstration program pans the color camera around a scene of colored objects, while simultaneously displaying the three top-ranked objects in order. Correct recognition is done at approximately 5 Hz. Swain has also developed a saliency measure that subtracts histogram features common to a known ensemble, thus weighting more heavily the features that are unique to each object. Last, there is a histogram backprojection algorithm that inverts the global, de-spaced nature of the histogram into scene locations that could have generated the histogram. Backprojection also runs at about 5 Hz, which allows the tracking of a moving object through a scene, or the location of a sought object when the camera moves smoothly or discontinuously over the scene.

1.2 Modeling attentional behavior sequences with an augmented hidden Markov model

Over the past year *Selective attention*, or the intelligent application of limited visual sensing and computing resources, has emerged as a basic topic for a long-range program of research we are now pursuing. The work in visual attention is proceeding in parallel with a new research direction in the Systems area - models and mechanisms for flexible, satisficing, reactive real-time computing.

Ray Rimey has implemented visual attention algorithms using a spatially-varying (foveal and peripheral) sensor. One aspect of the work attacks the specific problem of modeling *foveation sequences* [Rimey and Brown, 1990]. In most treatments of this subject, a sequence of eye movements emerges as a result of sequential cogni-

tive effort and image analysis, and is not explicitly represented. We augment the usual paradigm with a new explicit representation of probabilistic but task-dependent attentional sequencing. Explicit sequences are something like *motor skills*; they efficiently capture the effect of much cognitive activity and *feedback-mediated behavior*, and allow it to be generated quickly with low cognitive overhead.

The explicit representation is an augmented hidden Markov model (AHMM). A simple hidden Markov model can learn an emergent behavior and re-generate it as an explicit data-oblivious sequence. An AHMM incorporates a feedback sequence to modify the generated sequence (to pay attention to interesting things in unexpected places while following a scan path, for instance). It can therefore relearn or constantly modify its own explicit behavior, thus adapting to varying conditions. One AHMM model uses a simple external feedback loop, another uses internal feedback which modifies the internal parameters (probabilities) of the AHMM thus effecting the generation likelihoods directly. A third allows the foveation sequence to be expressed either in terms of image locations ("where") or image contents ("what") or both.

The Workbench for Active Vision Experimentation (WAVE) has now evolved into a general platform for integrating software. In anticipation of moving over to the Psyche operating system running on the Butterfly parallel computer, WAVE was converted to the g++ programming language used by Psyche and also was converted to use the Zebra system for programming our DataCube MaxVideo image processing hardware. WAVE was used to support the implementation of the selective attention model, and its performance is encouraging [Rimey and Brown, 1990].

1.3 Gaze Control and Segmentation

In research carried out at Oxford, Chris Brown did work on Kalman filters for tracking applications, cooperative work on projectively invariant matching of geometric structures in images [Brown *et al.*, 1989, Forsyth *et al.*, 1990] on control of Rochester's robot head [Brown 1989a,c; 1990a,b], and on computational properties of rotation representations [Brown, 1989b].

The control work investigated predictive mechanisms to solve problems of cooperation and delay. "Subsumption" architectures find these problems troublesome since internal state representations are minimized, control interaction is usually limited to preemption, and actions are synchronized only through the outside world. The work developed eight camera controls and investigated their interaction. It showed that predictive techniques can overcome the catastrophic effects of delays and interactions.

A fixated object "pops out" under ego motion since its surroundings blur. This method of pre-processing a scene to enhance edges and aid segmentation has parallels that use binocular visual capabilities, especially vergence. Coombs and Olson cooperated on vergence and segmentation algorithms for the robot head [Olson and Coombs, 1990] vergence has many advantages even

for systems without foveas. (1) **Mathematical simplification:** Fixating an object of interest puts points on the object near the optic axis in both eyes, allowing an orthographic projection model and simplifying many computations. (2) **Useful coordinate systems:** A unique fixation point defines a coordinate system that is related as much to the object being observed as it is to the observer, and hence is a step in the direction of an object-centered coordinate system [Ballard, 1989b]. (3) **Stereopsis:** Since the fixation point has a stereoscopic disparity of zero, points nearby will generally have small disparities. This makes it possible to use stereo algorithms that accept only a very limited range of disparities. Such systems can be very fast, and are amenable to hardware implementation. (4) **Disparity-based segmentation:** On the assumption that the gaze will normally be directed toward objects of interest, it may be appropriate for binocular agents to ignore features at large disparities. Thus disparity may be used to induce a segmentation on the scene.

The cepstral filter (akin to phase correlation) was used for the vergence calculations. It yields subpixel accuracy and, when used with a PD control law, is responsive to smooth and discontinuous variations in disparity. Vergence allows "zero disparity filtering" in which only zero-disparity points in images from the two eyes are passed on to further processing.

1.4 Parallel Cooperating Agents and Juggler

Juggler is a balloon-bouncing program under development, which has kept the balloon in the air for several seconds (some dozen hits). As of November 1989 [Yamauchi 1989], a version using five processors was running under the Psyche Operating System. The implementation uses binocular vision and a competing agent model of motor control; five processes compete with each other for access to the robot arm to position the balloon in the visual field, to position the racquet under the balloon, and to hit the balloon. Juggler is robust because even if processes had to share processors, failure to execute any one process during a particular time interval would have little if any affect on behavior: In the competing agent model, each application process continually broadcasts commands to the robot in competition with other processes. Our experiences with Juggler led to appropriate extensions to Psyche and communications capabilities, and we have now begun to design real-time facilities such as user-level scheduling.

1.5 Movement Detection

We have developed methods for the fast detection of moving objects from a platform that may itself be moving [Nelson 1990]. This task has applications in surveillance, process monitoring, and target detection. The primary challenge is to distinguish robustly the image motion due to independently moving objects from background flow induced by movement of the platform. Qualitative, pattern recognition strategies avoid the difficulties associated with quantitative determination of the image motion field.

Two complementary algorithms have been developed

and implemented. The first method, *constraint ray filtering*, uses knowledge about the observer's motion. It is based on the fact that, in a rigid environment, the projected 3-D velocity at any point in the image is constrained to lie on a 1-D locus in velocity space whose parameters depend only on the observer motion. Thus in principle, if the motion field and observer motion are known, an independently moving object can be detected because its projected velocity is unlikely to fall on this locus. In practice, quantitative estimates of the motion field and observer motion are both difficult and computationally expensive to obtain. Nelson adapts the basic principle to use partial information about the motion field and observer motion that can be rapidly and reliably computed from real image sequences.

The second method uses knowledge about the motion of the object to be detected. It takes advantage of the fact that the apparent motion of a fixed point due to smooth observer movement changes slowly while the apparent motion of moving objects such as animals or maneuvering vehicles often changes rapidly. Such *animate motion* can be detected by using the motion field at a given time to constrain the future motion field under smooth continuation, and then looking for violations of these constraints.

Both methods are implemented using the Datacube Maxvideo system for low-level qualitative motion extraction and a SUN workstation for higher-level processing. The algorithms run in real time (10 Hz, 1/10 second latency) and successfully detect independent movement from a moving platform in a variety of situations, at full (512 × 512) resolution. Details can be found in the paper "Qualitative Detection of Motion by a Moving Observer" in these proceedings.

2 Computing Environments for Parallel Vision

2.1 Languages and Operating Systems

An alternate communications library for the Puma robot (ROBOCOM) was written by Brian Yamauchi and John Soong for use in the Juggler project. ROBOCOM is much faster than the BOTLIB package since it does not use the multi-layered ISO-standard structure for communication.

This year Rochester released Zebra, an object oriented programming interface to Datacube's MaxVideo family of image processing boards. Each board type is represented by an object class. Each physical MaxVideo board is represented by an instance of its class. Simply by declaring the board objects as variables, the boards are opened and initialized. Zebra takes a microprogramming-like approach to controlling Datacube boards. The register set for each board is considered to be a microinstruction word. This instruction word completely specifies a board configuration. By sending instruction words to boards, the hardware can be completely programmed in a microprogramming-like manner. Instruction words can be stored in and retrieved from files, allowing the sharing of standard configurations between developers. Instruction words are created and modified via an instruc-

tion word editor. One such editor is Zed, a graphical tool provided with Zebra that drastically reduces the learning curve for Datacube programming [Tilley, 1990].

Over the past year several languages for MIMD parallel computers have been developed and ported, and quantitative comparisons made between programming models. Parallel compilation issues and programming paradigms are being explored in the thesis work of Gafter and Crowl [Crowl, 1989b]. The key reports are [Baldwin 1989a,b,c (Consul); Scott et al. 1990a (Multi-model parallel programming); Crowl 1989b (A uniform object model); Tilley 1990 (Zebra for MaxVideo)].

Three operating systems (Elmwood, Platinum, Psyche) have been developed for the Butterfly. The most ambitious project is Psyche, though Platinum and Osmium solve automatically a number of problems that users face when using Uniform System-style programming on a MIMD computer (Automatic cacheing and data migration, for instance). The key papers are [Scott et al. 1989b,c (Psyche description); LeBlanc et al. 1989b (Elmwood description); Cox and Fowler 1989 (Platinum description)]

We believe that building an integrated, reasoning, reactive vision and robotic system requires multiple models of parallel computation. Psyche provides a low-level interface with uniform naming and an emphasis on dynamic fine-grained sharing. Through its use of data abstraction, lazy evaluation of protection, and parameterized user-level scheduling, it allows programs written under many different programming models to coexist and interact. The conventions of realm protocols, upcalls, and block and unblock routines provide a structure for communication across models that is, to the best of our knowledge, unprecedented. With appropriate permissions, user-level code can exercise full control over the physical resources of memory, processors, and devices. In effect, it should be possible under Psyche to implement almost any application for which the underlying hardware is appropriate. This, for us, constitutes the definition of "general-purpose parallel computing."

Psyche differs from existing multiprocessor operating systems in several fundamental ways.

1. It employs a uniform name (address) space for all its user programs without relying on compiler support for protection.
2. It evaluates access rights lazily, permitting the distribution of rights without kernel intervention.
3. It places the management of threads, and in fact their definition, in the hands of user-level code.
4. It minimizes the need for kernel calls in general by relying whenever possible on shared user/kernel data structures that can be examined asynchronously.
5. It provides the user with an explicit tradeoff between protection and performance by facilitating the interchange of protected and optimized invocations.

As of November 1989 we were able to run our first real user applications. Implemented portions of Psyche

include low-level machine support. interrupt handlers, virtual memory (without paging), full support for inter-kernel shared memory, synchronous inter-kernel communication via remote interrupts, support for atomic hardware operations, remote source-level kernel debugging, and loading of the kernel via Ethernet. Also core support for the Psyche user interface: realms, virtual processors, protection domains, keys and access lists, software interrupts, and protected and optimized invocation of realm operations. Also rudimentary I/O to the console serial device, and remote file service via Ethernet. There are minimal user-level tools: a simple shell, program loader, and name server, support for command-line argument passing, simple handlers for software interrupts, and standard I/O and kernel call libraries.

2.2 Performance Monitoring and Debugging

This year we honed several of our tools to help the user effectively implement parallel algorithms [e.g. LeBlanc 1989; LeBlanc et al. 1990; Mellor-Crummey 1989b]. The main thrust has been the construction of parallel performance monitoring tools and experimentation with the use of these tools [e.g. Fowler and Bella 1989; Fowler et al. 1989].

The information collected during program monitoring can be used to replay a program during the debugging cycle. During replay, events can be observed at any level of detail, and controlled experiments can be performed. More important, however, is the use of program monitoring to create a representation for an execution that can be analyzed by our programmable toolkit.

The core of our toolkit consists of facilities for recording execution histories, a common user interface for the interactive, graphical manipulation of those histories, and tools for examining and manipulating program state during replay of a previously recorded execution. The user interface for the toolkit resides on the programmer's workstation and consists of two major components: an interactive, graphical browser for analyzing execution histories, and a programmable Lisp environment. The execution history browser, called Moviola, is written in C and runs under the X Windows System.

Moviola implements a graphical view of an execution based on a DAG representation of processes and communication. In a Moviola diagram, time flows from top to bottom. Events that occur within a process are aligned vertically, forming a time-line for that process. Edges joining events in different processes reflect temporal relationships resulting from synchronization. Event placement is determined by global logical time computed from the partial order of events collected during execution. Each event is displayed as a shaded box with height proportional to the duration of the event, and boxes are connected with lines representing interactions. Irregularity, inactive processors, and other indications of possible bugs are readily apparent in such a diagram.

We have successfully used this facility for kernel debugging and plan to use it as a base for user-level, multi-model debugging. Low-level debugger functions will be implemented by a combination of gdb and lld. High-level commands from the user will be translated by a

model-specific interface, created as part of the programming model. The Moviola graphical interface has been improved, significantly reducing the display time and increasing the functionality. The S graphics package has been added to the toolkit, facilitating graphical displays of performance data. LISP tools have been written for critical path analysis and for gathering and plotting performance statistics.

3 Related Theses, 1989-1990

Yap, Sue-Ken., "PENGUIN: A language for reactive graphical user interfaces": Technical Report 344 (and Ph.D. Thesis, April 1990).

PENGUIN (Programming Environment for Graphical User Interfaces) is a computer language that supports grammar-based specification of control flow in event-driven graphical programs. The PENGUIN model of intercomponent connection extends and subsumes the older Seeheim model of UIMS design, allowing large programs to be constructed as co-operating components. If the reactive nature of graphical programs should be taken into account from the beginning of design, a graphical program can be composed as a collection of modules whose input behaviour is specified, and modules be grouped into separately-compiled components along lines of clear division of labour and responsibility for resources. Such partitions result in components that are more likely to be reusable. Our experiences indicate that the use of PENGUIN can reduce the volume of user interface code by a factor of two to three and result in code which is clearer than functionally equivalent code using traditional control structures. Uniform handling of I/O and signals as PENGUIN events leads to programs that are more portable across systems.

Gafter, N., "Parallel Incremental Compilation," Ph.D. Thesis, June 1990: We describe a set of techniques that enable incremental compilation to exploit fine-grained concurrency in a shared-memory multiprocessor and achieve asymptotic improvement over sequential algorithms. Because parallel non-incremental compilation is a special case of parallel incremental compilation, the design of a parallel compiler is a corollary of our result. Instead of running the individual phases concurrently, our design specifies compiler phases that are mutually sequential. However, each phase is designed to exploit fine-grained parallelism. By allowing each phase to present its output as a complete structure rather than as a stream of data, we can apply techniques such as parallel prefix and parallel divide-and-conquer, and we can construct applicative data structures to achieve sublinear execution time. We describe new algorithms for parsing using a balanced list representation and type checking based upon attribute grammars modified with a combination of aggregate values and upward remote references. Under some mild assumptions about the language and target program, these phases run in polylogarithmic time using a sublinear number of processors.

Dibble, P.C., "A Parallel Interleaved File System," Ph.D. Thesis and TR 334, March 1990. This dissertation introduces the concept of a parallel interleaved file system. This class of file system incorporates three con-

cepts: parallelism, interleaving, and tools. Parallelism appears as a characteristic of the file system program and in the disk hardware. The parallel file system software and hardware allows the file system to scale with the other components of a multiprocessor computer. Interleaving is the rule the file system uses to distribute data among the processors.

Floyd, R.A., "Transparency in distributed file systems," Ph.D. Thesis and TR 272, January 1989: Our distributed file system, Roe, supports a substantially higher degree of transparency than earlier distributed file systems, and is able to do this in a heterogeneous environment. Roe appears to users to be a single, globally accessible file system providing highly available, consistent files. It provides a coherent framework for uniting techniques in the areas of naming, replication, consistency control, file and directory placement, and file and directory migration in a way that provides full network transparency.

Mellor-Crummey, J., "Debugging and analysis of large-scale parallel programs," Ph.D. Thesis and TR 312, September 1989: This dissertation addresses the problem of debugging and analysis of large-scale parallel programs executing on shared-memory multiprocessors. It is shown how synchronization traces can be used to create indistinguishable executions that form the basis for debugging. This result is used to develop a practical technique for tracing parallel program executions on shared-memory parallel processors so that their executions can be repeated deterministically on demand. The design of an integrated, extensible toolkit based on these traces is proposed. This toolkit uses execution traces to support interactive, graphics-based, top-down analysis of parallel program executions.

Olson, T.J., "An architectural model of visual motion understanding," Ph.D. Thesis and TR 305, August 1989: The central claim of this thesis is that many puzzling aspects of motion perception can be understood by assuming a particular architecture for the human motion processing system. The architecture consists of three functional units or subsystems. The first or low-level subsystem computes simple mathematical properties of the visual signal. It is entirely bottom-up, and prone to error when its implicit assumptions are violated. The intermediate-level subsystem combines the low-level system's output with world knowledge, segmentation information and other inputs to construct a representation of the world in terms of primitive forms and their trajectories. It is claimed to be the substrate for long-range apparent motion. The highest level of the motion system assembles intermediate-level form and motion primitives into scenarios that can be used for prediction and for matching against stored models. Simulation results show that its interpretations are in qualitative agreement with human perception.

Cooper, P.R., "Parallel object recognition from structure (The Tinkertoy project)," Ph.D. Thesis and TR 301, July 1989. The task is the recognition of objects whose identity is defined solely by the spatial relationships between simple parts. A massively parallel framework incorporating a principled treatment of uncertainty

and domain dependence is developed to address the problem. The basic architecture of the solution is formed by posing structure matching as a part-wise correspondence problem in a labelling framework, and then applying the unit/value principle. The solution is developed incrementally. Complexity and correctness analyses and implementation experiments are provided at each phase. The formulation of the application problem is also generalized, so geometric discrimination can be achieved. The solution is generalized to handle uncertain input information and statistical domain dependence. Segmentation and recognition are computed simultaneously by a coupled Markov Random Field. The method deals well with occlusion and accidental alignment.

Swain, Michael, "Color Indexing", Ph.D. thesis, June 1990. This dissertation demonstrates that color histograms of multicolored objects provide a robust, efficient cue for indexing into a large database of models. It shows that color histograms are stable object representations in the presence of occlusion and over change in view, and that they can differentiate among a large number of objects. For solving the identification problem, it introduces a technique called *Histogram Intersection*, which matches model and image histograms and a fast incremental version of Histogram Intersection that allows real-time indexing into a large database of stored models. It demonstrates techniques for dealing with crowded scenes and with models with similar color signatures. For solving the location problem it introduces an algorithm called *Histogram Backprojection* that performs this task efficiently in crowded scenes.

References

- [Aloimonos and Brown, 1989] John Aloimonos and Christopher M. Brown. On the kinetic depth effect. *Biological Cybernetics*, 60(6):445-455, 1989.
- [Baldwin, 1989a] Douglas Baldwin. CONSUL: A parallel constraint language. *IEEE Software*, pages 62-69, July 1989.
- [Baldwin, 1989b] Douglas Baldwin. Preliminary estimates of parallelism in CONSUL programs. In *Proc., 22nd Hawaii Int'l. Conf. on System Sciences*, Kona, HI, January 1989.
- [Baldwin, 1989c] Douglas Baldwin. A status report on CONSUL. In *Proc., 2nd Workshop on Programming Languages and Compilers for Parallel Programming*. Univ. of Illinois, August 1989.
- [Ballard et al., 1989] Dana H. Ballard, Randal C. Nelson, and Brian Yamauchi. Animate vision. *Optics News*, 15(5), May 1989. Invited article; also appeared in *1989-90 Computer Science and Engineering Research Review*, Computer Science Dept., Univ. of Rochester, December 1989.
- [Ballard et al., 1990] Dana H. Ballard, Lambert E. Wixson, and Michael J. Swain. Animate vision and object search. in preparation, Department of Computer Science, University of Rochester, June 1990.
- [Ballard, 1989a] Dana H. Ballard. Behavioral constraints on animate vision. *Image and Vision Computing*, 7(1):3-9, February 1989. Also appeared in *Proc., 4th ALVEY Vision Conference*, Sept. 1988.
- [Ballard, 1989b] Dana H. Ballard. Reference frames for animate vision. In *Proc., International Joint Conference on Artificial Intelligence*, pages 1635-1641, Detroit, MI, August 1989. AAAI. Also appeared in *Proc., 2nd Int'l. Congress on Neuroethology*, Berlin, Sept. 1989.
- [Ballard, 1990a] Dana H. Ballard. Animate vision. Technical Report 329, Department of Computer Science, University of Rochester, February 1990. Also to appear, *AI Journal*.
- [Ballard, 1990b] Dana H. Ballard. Animate vision uses object-centered reference frames. In *Int'l. Symp. on Neural Networks for Sensory and Motor Systems (NSMS)*, Düsseldorf, West Germany, March 1990.
- [Ballard, to appear] Dana H. Ballard. Eye movements and spatial cognition. Forthcoming book (*Proc., 1988 Workshop on Exploratory Vision: The Pictive Eye*). Department of Computer Science, University of Rochester, to appear. Also appeared as TR 218, Computer Science Dept., Univ. of Rochester, Nov. 1987; and in the *AAAI Spring Symp. Series on Physical and Biological Approaches to Computational Vision*, March 1988.
- [Bandopadhyay and Ballard, to appear] Amit Bandopadhyay and Dana H. Ballard. Active navigation. Egomotion perception by the tracking observer. *Computational Intelligence*, to appear.
- [Bolosky et al., 1989] William J. Bolosky, R.P. Fitzgerald, and Michael L. Scott. Simple but effective techniques for NUMA memory management. In *Proc., Twelfth ACM Symp. on Operating Systems Principles*, pages 19-31, Litchfield Park, AZ, December 1989. Also appeared in *ACM Operating Systems Review*, 23(5).
- [Brown and Nelson, 1989] Christopher M. Brown and Randal C. Nelson. Image understanding at the University of Rochester. In *Proc., DARPA Image Understanding Workshop*, Palo Alto, CA, May 1989.
- [Brown and others, 1989] Christopher M. Brown et al. Three-dimensional vision techniques for autonomous vehicles. In R.C. Jain and A.K. Jain, editors, *Analysis and Interpretation of Range Images*. Springer-Verlag, New York, 1989.
- [Brown et al., 1989] Christopher M. Brown, Hugh F. Durrant-Whyte, John Leonard, Bobby Rao, and Barry Steer. Centralized and decentralized kalman filter techniques for tracking, navigation, and control. Technical Report 277 (revised), Department of Computer Science, University of Rochester, May 1989. Also appeared as Report OUEL 1765/89, Robotics Research Group, Dept. of Engg. Science, Univ. of Oxford, May 1989; and *Proc., DARPA Image Understanding Workshop*, 651-675, Palo Alto, CA, May 1989.
- [Brown, 1989a] Christopher M. Brown. Gaze behaviors for robotics. In *Proc., NATO-ACI Symp. on Active*

- Perception and Robot Vision*, Maratea, Italy, July 1989. Invited paper.
- [Brown, 1989b] Christopher M. Brown. Some computational properties of rotation representations. Technical Report 303 (revised), Department of Computer Science, University of Rochester, August 1989.
- [Brown, 1989c] Christopher M. Brown. Three-dimensional dynamic and kinematic prediction in gaze control. In *Proc., Workshop on Interpretation of 3D Scenes*, Austin, TX, November 1989. IEEE.
- [Brown, 1990a] Christopher M. Brown. Gaze controls cooperating through prediction. *Image and Vision Computing*, 8(1):10-17, February 1990. Special edition.
- [Brown, 1990b] Christopher M. Brown. Gaze controls with interactions and delays. *IEEE Transactions on Systems, Man, and Cybernetics*, IEEE-TSMC20(3), May 1990. Also appeared as TR 278, Computer Science Dept., Univ. of Rochester, March 1989; Report OUEL 1770/89, Robotics Research Group, Dept. of Eng. Science, Univ. of Oxford, March 1989; and *Proc., DARPA Image Understanding Workshop*, Palo Alto, CA, May 1989.
- [Brown, 1990 in press] Christopher M. Brown. Prediction and cooperation in gaze control. *Biological Cybernetics*, May 1990, in press. Also appeared as TR 295, Computer Science Dept., Univ. of Rochester, May 1989; and Report OUEL 1771/89, Robotics Research Group, Dept. of Eng. Science, Univ. of Oxford, May 1989.
- [Chou and Brown, 1990] Paul B. Chou and Christopher M. Brown. The theory and practice of Bayesian image labeling. *International Journal of Computer Vision*, 4(3):185-210, June 1990. Also appeared as IBM Research Rpt. RC 15460, T.J. Watson Research Center, Feb. 1990.
- [Coombs, 1989a] David J. Coombs. Gaze stabilization for animate vision. Thesis proposal, Department of Computer Science, University of Rochester, December 1989.
- [Coombs, 1989b] David J. Coombs. Tracking objects with eye movements. In *Topical Meeting on Image Understanding and Machine Vision*. Optical Society of America, June 1989. Also appeared in *Proc., 4th Annual Univ. of Buffalo Graduate Conf. on Computer Science*, Computer Science Dept., SUNY Buffalo, March 1989.
- [Cooper and Swain, 1989] Paul R. Cooper and Michael J. Swain. Domain dependence in parallel constraint satisfaction. In *Proc., International Joint Conference on Artificial Intelligence*, Detroit, MI, August 1989.
- [Cooper, 1989] Paul R. Cooper. *Parallel Object Recognition from Structure (The Tinkertoy Project)*. PhD thesis, Department of Computer Science, University of Rochester, July 1989. Also appeared as TR 301, July 1989.
- [Cox and Fowler, 1989] Alan S. Cox and Robert J. Fowler. The implementation of a coherent memory abstraction on a NUMA multiprocessor: Experiences with platinum. In *Proc., 12th ACM Symp. on Operating System Principles*, pages 32-44, Litchfield Park, AZ, December 1989.
- [Crowl, 1989a] Lawrence A. Crowl. Concurrent data structures and actor programming under the matroshka model. *ACM SIGPLAN Notices*, 24(4):79-80, April 1989. Also appeared in *Proc., ACM SIGPLAN Workshop on Concurrent Object-Based Programming*, 79-80, Sept. 1988.
- [Crowl, 1989b] Lawrence A. Crowl. A uniform object model for parallel programming. *ACM SIGPLAN Notices*, 24(4):25-27, April 1989. Also appeared in *Proc., ACM SIGPLAN Workshop on Concurrent Object-Based Programming*, 25-27, Sept. 1988.
- [Dibble and Scott, 1989a] Peter C. Dibble and Michael L. Scott. Beyond striping: The bridge multiprocessor file system. *Computer Architecture News*, 17(5):32-39, September 1989.
- [Dibble and Scott, 1989b] Peter C. Dibble and Michael L. Scott. External sorting on a parallel interleaved file system. In *1989-90 Computer Science and Engineering Research Review*, pages 20-27. Department of Computer Science, University of Rochester, December 1989.
- [Dibble and Scott, 1990] Peter C. Dibble and Michael L. Scott. The parallel interleaved file system: A solution to the multiprocessor I/O bottleneck. Technical report, Department of Computer Science, University of Rochester, May 1990. Submitted for publication.
- [Dibble, 1990] Peter C. Dibble. *A Parallel Interleaved File System*. PhD thesis, Department of Computer Science, University of Rochester, March 1990. Also appeared as TR 334, March 1990.
- [Feist, 1989a] Steven E. Feist. Integrating symbolic planning and reactive execution. Thesis proposal, Department of Computer Science, University of Rochester, August 1989.
- [Feist, 1989b] Steven E. Feist. Moving towards reactivity in planning systems. Internal report, Department of Computer Science, University of Rochester, January 1989.
- [Finkel et al., 1989] R.A. Finkel, M.L. Scott, Y. Artsy, and H.-Y. Chang. Experience with Charlotte: Simplicity and function in a distributed operating system. *IEEE Transactions on Software Engineering*, 15(6):676-685, June 1989. Extended abstract presented at the IEEE Workshop on Design Principles for Experimental Distributed Systems, Purdue University, 15-17 October 1986; earlier full-length version appeared as Computer Sciences TR 653, U. Wisconsin, Madison, July 1986.
- [Floyd, 1989] Richard A. Floyd. *Transparency in Distributed File Systems*. PhD thesis, Department of Computer Science, University of Rochester, January 1989. Also appeared as TR 272, January 1989.

- [Forsyth et al., 1990] D. Forsyth, J.L. Mundy, A. Zisserman, and C. Brown. Projectively invariant representations using implicit algebraic curves. In *Proceedings of the First European Vision Conference*, 1990.
- [Fowler and Bella, 1989] Robert J. Fowler and Ivan Bella. A programmer's guide to Moviola: An interactive execution history browser. Technical Report 269, Department of Computer Science, University of Rochester, February 1989.
- [Fowler et al., 1989] Robert J. Fowler, Thomas J. LeBlanc, and John M. Mellor-Crummey. An integrated approach to parallel program debugging and performance analysis on large-scale multiprocessors. *ACM SIGPLAN Notices*, 24(1):163-173, January 1989. Also appeared in *Proc., ACM SIGPLAN/SIGOPS Workshop on Parallel and Distributed Debugging*, 163-173, Madison, WI, May 1988.
- [Hartman, to appear 1990] Leo B. Hartman. *Decision Theory and the Cost of Planning*. PhD thesis, Department of Computer Science, University of Rochester, to appear, 1990.
- [Hayhoe et al., 1990] Mary Hayhoe, P. Moeller, Dana H. Ballard, and Joanne E. Albano. Guidance of saccades to remembered targets and the perception of spatial position. *Investigative Ophthalmology and Visual Science Supplement*, 31:603, 1990.
- [LeBlanc and Miller, 1989] Thomas J. LeBlanc and B.P. Miller (Eds.). Edited Summary of the ACM SIGPLAN/SIGOPS Workshop on Parallel and Distributed Debugging. *SIGPLAN Notices*, 24(1):ix-xxi, January 1989. Also appeared in *Operating Systems Review* 22, 4, 7-19, Oct. 1988.
- [LeBlanc and Markatos, 1990] Thomas J. LeBlanc and Evangelos P. Markatos. Operating system support for adaptable real-time systems. In *Proc., Seventh IEEE Workshop on Real-Time Operating Systems and Software*, pages 1-10, Charlottesville, VA, May 1990.
- [LeBlanc and Mellor-Crummey, to appear 1990] Thomas J. LeBlanc and John M. Mellor-Crummey. Debugging parallel programs with instant replay. In P.J. Waterman, editor, *Applications on the Butterfly Parallel Processor. Research Monographs on Parallel and Distributed Computing*. Pitman Publishing / MIT Press, London, to appear, 1990.
- [LeBlanc et al., 1989a] Thomas J. LeBlanc, Brian D. Marsh, and Michael L. Scott. Memory management for large-scale NUMA multiprocessors. Technical Report 311, Department of Computer Science, University of Rochester, March 1989.
- [LeBlanc et al., 1989b] Thomas J. LeBlanc, John M. Mellor-Crummey, Neal M. Gafter, Lawrence A. Crowl, and Peter C. Dibble. The Elmwood multiprocessor operating system. *Software—Practice and Experience*, 19(11):1029-1056, November 1989.
- [LeBlanc et al., to appear 1990] Thomas J. LeBlanc, John M. Mellor-Crummey, and Robert J. Fowler. Analyzing parallel program executions using multiple views. *Journal of Parallel and Distributed Computing*, to appear, 1990.
- [LeBlanc, 1989] Thomas J. LeBlanc. Parallel program debugging. In *Proc., 13th Annual IEEE Int'l. Computer Software and Applications Conf. (COMPSAC 89)*, Orlando, FL, September 1989.
- [LeBlanc, to appear 1990] Thomas J. LeBlanc. Structured message passing on a shared-memory multiprocessor. In P.J. Waterman, editor, *Applications on the Butterfly Parallel Processor. Research Monographs on Parallel and Distributed Computing*. Pitman Publishing / MIT Press, London, to appear, 1990.
- [Martin and Allen, 1990] Nathaniel G. Martin and James F. Allen. Abstraction in planning: A probabilistic approach. Technical report, Department of Computer Science, University of Rochester, 1990. Submitted for conference publication.
- [Martin et al., 1990] Nathaniel G. Martin, James F. Allen, and Christopher M. Brown. Armtrak: A domain for the unified study of natural language, planning, and active vision. Technical Report 324 (revised), Department of Computer Science, University of Rochester, January 1990.
- [Mellor-Crummey and LeBlanc, 1989] John M. Mellor-Crummey and Thomas J. LeBlanc. A software instruction counter. In *Proc., 3rd Int'l. Conf. on Architectural Support for Programming Languages and Operating Systems*, pages 78-86, Boston, MA, April 1989.
- [Mellor-Crummey and Scott, 1990] John M. Mellor-Crummey and Michael L. Scott. Algorithms for scalable synchronization on shared-memory multiprocessors. Technical Report 342, Department of Computer Science, University of Rochester, April 1990. Also appeared as TR90-114, Center for Research on Parallel Computation, Rice University.
- [Mellor-Crummey, 1989a] John M. Mellor-Crummey. The 1988 Int'l. Conf. on Fifth Generation Computer Systems: A trip report. *Scientific Information Bulletin*, 14(4), October-December 1989.
- [Mellor-Crummey, 1989b] John M. Mellor-Crummey. *Debugging and Analysis of Large-Scale Parallel Programs*. PhD thesis, Department of Computer Science, University of Rochester, September 1989. Also appeared as TR 312, Sept. 1989.
- [Nelson and Aloimonos, 1989] Randal C. Nelson and John Aloimonos. Obstacle avoidance using flow field divergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(10):1102-1106, October 1989.
- [Nelson et al., 1989] Randal C. Nelson, Dana H. Ballard, and Steven D. Whitehead. Visual behavior and intelligent agents. In *Proc., SPIE Symp. on Advances in Intelligent Robotics Systems*, Philadelphia, PA, November 1989.
- [Nelson, 1989] Randal C. Nelson. Visual homing using an associative memory. In *Proc., DARPA Image Understanding Workshop*, Palo Alto, CA, May 1989. Also

- appeared in *AAAI 1989 Spring Symp. Series*, Stanford, CA, March 1989; and submitted for journal publication.
- [Nelson, 1990] Randal C. Nelson. Qualitative detection of motion by a moving observer. Technical Report 341, Department of Computer Science, University of Rochester, April 1990. Submitted for conference and journal publication.
- [Olson and Coombs, 1990] T.J. Olson and D.J. Coombs. Real-time vergence control for binocular robots. *submitted for publication*, 1990.
- [Olson and Potter, 1989] Thomas J. Olson and Robert D. Potter. Real-time vergence control. In *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, pages 404-409, San Diego, CA, June 1989. IEEE. Also appeared as TR 264, Computer Science Dept., Univ. of Rochester, Nov. 1988.
- [Olson, 1989] Thomas J. Olson. *An Architectural Model of Visual Motion Understanding*. PhD thesis, Department of Computer Science, University of Rochester, August 1989. Also appeared as TR 305, August 1989.
- [Rimey and Brown, 1990] Ray D. Rimey and Christopher M. Brown. Selective attention as sequential behavior: Modeling eye movements with an augmented hidden markov model. Technical Report 327 (revised), Department of Computer Science, University of Rochester, April 1990. Submitted for publication.
- [Scott and Jones, to appear 1990] Michael L. Scott and Kurt R. Jones. Ant Farm: A lightweight process programming environment. In P.J. Waterman, editor, *Applications on the Butterfly Parallel Processor*. Pitman Publishing / MIT Press, London, to appear, 1990. Also appeared as BPR 21, Computer Science Dept., Univ. of Rochester, Aug. 1988.
- [Scott et al., 1989a] Michael L. Scott, Thomas J. LeBlanc, and Brian D. Marsh. Evolution of an operating system for large-scale shared-memory multiprocessors. Technical Report 309, Department of Computer Science, University of Rochester, March 1989.
- [Scott et al., 1989b] Michael L. Scott, Thomas J. LeBlanc, and Brian D. Marsh. Implementation issues for the Psyche multiprocessor operating system. In *Proc., First Workshop on Experiences With Building Distributed and Multiprocessor Systems*, pages 227-236, Ft. Lauderdale, FL, October 1989.
- [Scott et al., 1989c] Michael L. Scott, Thomas J. LeBlanc, and Brian D. Marsh. A multi-user, multi-language open operating system. In *Proc., Second Workshop on Workstation Operating Systems*, pages 125-129, Pacific Grove, CA, September 1989.
- [Scott et al., 1990a] Michael L. Scott, Thomas J. LeBlanc, and Brian D. Marsh. Multi-model parallel programming in Psyche. In *Proc., Second ACM SIGPLAN Symp. on Principles and Practice of Parallel Programming (PPoPP)*, pages 70-78, Seattle, WA, March 1990.
- [Scott et al., 1990b] Michael L. Scott, Thomas J. LeBlanc, Brian D. Marsh, Timothy G. Becker, Cezary Dubnicki, Evangelos P. Markatos, and Neil G. Smithline. Implementation issues for the Psyche multiprocessor operating system. *USENIX Computing Systems Journal*, 3(1), Spring 1990. Earlier version presented at the First Workshop on Experiences with Building Distributed and Multiprocessor Systems, Ft. Lauderdale, FL, October 1989, 227-236.
- [Scott, 1989] Michael L. Scott. An overview of Lynx. Technical Report 308, Department of Computer Science, University of Rochester, August 1989. Revised version submitted for publication, February 1990.
- [Shaffer et al., to appear] C.A. Shaffer, H. Samet, and Randal C. Nelson. QUILT: A geographic information system based on quadrees. *Int'l. Journal of Geographic Information Systems*, to appear.
- [Simard and Mailloux, to appear 1990] Patrice Y. Simard and G.E. Mailloux. Vector field restoration by the method of convex projections. *Computer Vision, Graphics, and Image Processing*, to appear, 1990.
- [Simard, to appear 1990] Patrice Y. Simard. Using recurrent networks to learn sequences over time. *Neural Network Review*, to appear, 1990. Invited critical review.
- [Swain and Ballard, 1989] Michael J. Swain and Dana H. Ballard. Indexing via color histograms. In *Proc., NATO Workshop on Computer Vision*, Maratea, Italy, July 1989.
- [Swain and Wixson, 1989] Michael J. Swain and Lambert E. Wixson. Efficient estimation for markov random fields. In *Proc., Optical Society of America Topical Meeting on Image Understanding and Machine Vision*, N. Falmouth, MA, June 1989.
- [Swain et al., to appear 1990a] Michael J. Swain, Lambert E. Wixson, and Dana H. Ballard. Object identification and search: Animate vision alternatives to image interpretation. In Paolo Dario, Giulio Sandini, and Patrick Aebischer, editors, *Robots and Biological Systems*. Springer Verlag, to appear, 1990. *Proc., NATO Advanced Research Workshop on Robots and Biological Systems*, June 1989.
- [Swain et al., to appear 1990b] Michael J. Swain, Lambert E. Wixson, and Paul B. Chou. Efficient parallel estimation for Markov random fields. In M. Henrion, R. Schacter, L. N. Kanal, and J. Lemmer, editors, *Uncertainty in Artificial Intelligence: Volume V*. North-Holland, to appear, 1990. Also appeared in *Proc., Fifth Workshop on Uncertainty and Artificial Intelligence*, 361-368, Windsor, Ontario, August 1989.
- [Tilley, 1990] David G. Tilley. Zebra for MaxVideo: An application of object oriented microprogramming to register level devices. Technical Report 315, Department of Computer Science, University of Rochester, February 1990.
- [Weber, 1989] Jay C. Weber. A parallel algorithm for statistical belief refinement and its use in causal rea-

- soning. In *Proc., International Joint Conference on Artificial Intelligence*, August 1989.
- [Whitehead and Ballard, 1989a] Steven D. Whitehead and Dana H. Ballard. Reactive behavior, learning, and anticipation. In *1989 NASA Conf. on Space Tele-robotics*, Pasadena, CA, February 1989.
- [Whitehead and Ballard, 1989b] Steven D. Whitehead and Dana H. Ballard. A role for anticipation in reactive systems that learn. In *Proc., 6th Int'l. Workshop on Machine Learning*, Cornell University, Ithaca, NY, June 1989.
- [Whitehead and Ballard, 1990] Steven D. Whitehead and Dana H. Ballard. Learning to perceive and act. In *Proc., Machine Learning Conference*, June 1990. Also appeared as TR 331 (revised), Computer Science Dept., Univ. of Rochester, June 1990.
- [Whitehead, 1989] Steven D. Whitehead. Thesis proposal: Scaling reinforcement learning systems. Technical Report 304, Department of Computer Science, University of Rochester, August 1989.
- [Wixson and Ballard, 1990] Lambert E. Wixson and Dana H. Ballard. Real-time detection of multi-colored objects. In P.S. Schenker, editor, *Sensor Fusion II: Human and Machine Strategies*, pages 435-446. SPIE, 1990. Volume 1198.
- [Wixson, 1990b] Lambert E. Wixson. The acquisition and utilization of high-level knowledge about spatial relationships to search for an object. Technical Report 338, Department of Computer Science, University of Rochester, April 1990.
- [Yamauchi, 1989] Brian Yamauchi. Juggler: Real-time sensorimotor control using independent agents. In *Proc., Optical Society of America Image Understanding and Machine Vision Conf.*, N. Falmouth, MA, June 1989. Image Understanding and Machine Vision, 1989 Technical Digest Series, Vol. 14, Optical Society of America, Washington, DC, June 1989.
- [Yamauchi, 1990a] Brian Yamauchi. An evolutionary approach to building autonomous robots. In *Presentation, 2nd Artificial Life Workshop*, Santa Fe, NM, February 1990. Santa Fe Institute / Los Alamos Center for Nonlinear Studies.
- [Yamauchi, 1990b] Brian Yamauchi. Independent agents: A behavior-based architecture for autonomous robots. In *5th Annual Graduate Conf. on Computer Science*, SUNY Buffalo, Buffalo, NY, March 1990.
- [Yap and Scott, 1990] Sue-Ken Yap and Michael L. Scott. PENGUIN: A language for reactive graphical user interface programming. submitted for publication, Department of Computer Science, University of Rochester, February 1990.

Progress in Computer Vision at the University of Massachusetts¹

Edward M. Riseman and Allen R. Hanson
Computer Vision Research Laboratory
Dept. of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

ABSTRACT¹

This report summarizes progress in image understanding research at the University of Massachusetts over the past year. Many of the individual efforts discussed in this paper are further developed in other papers in this proceedings. The summary is organized into several areas:

1. Mobile Robot Navigation
2. Motion and Stereo Processing
3. Knowledge-Based Interpretation of Static Scenes
4. Image Understanding Architecture

The research program in computer vision at UMass has as one of its goals the integration of a diverse set of research efforts into a system that is ultimately intended to achieve real-time image interpretation in a variety of vision applications.

1. Mobile Robot Navigation

The initial focus of the mobile robot navigation project (Fennema and Hanson 1990b) has been on the development of a system for goal oriented navigation through a partially modeled, unchanging environment which contains no unmodeled obstacles. This simplified environment is intended to provide a foundation for research into navigation in more complicated domains. The guiding philosophy of this project is a tight coupling between planning, perception, and plan execution. Incremental planning and vehicle motion, guided by the relationship between the internal model and the external world provided by perception, serve to keep the vehicle accurately located within the environmental reference frame.

1.1 Experiments in Planning and Plan Execution

In a recent experiment (Fennema and Hanson 1990b), the vehicle successfully navigated a multi-leg course, from the robot laboratory to an office, moving approximately 50 feet

through two doorways and coming to rest within an inch of its intended goal. Both rooms and the hallway were accurately (but incompletely) modelled using a hierarchical volumetric representation of space. Volumes are represented by their surrounding faces, and faces contain information about the visual appearance of the face. A more complete description of the world representation may be found in (Fennema and Hanson 1990a; Fennema and Hanson 1990b). Clearly, many more experiments, under widely varying environmental conditions (both indoors and outdoors), must be run before the robustness of the techniques can be established.

1.1.1 Plan Generation

Planning is carried out over the world model using traditional planning techniques (e.g. A* search, freespace representations, etc.) to generate a sequence of plan 'sketches' (incompletely specified plans). Plans are generated in a depth first manner, with more detailed plans closer to the vehicle's current location. Associated with each plan sketch is a milestone, which can be thought of as a precondition for the execution of the plan sketch. These milestones are typically specified as landmarks which must be verified visually (and the vehicle's position relative to them determined) prior to the execution of the next step in the plan. The milestones form the basis for 'plan-level servoing', discussed below.

1.1.2 Plan Execution and Perceptual Servoing

The rationale for not fully developing detailed plans prior to moving the vehicle is that plans fail. Obstacles in the planned path, irregular or slippery surfaces, uneven tire inflation, or unexpected externally induced vehicle motions can throw the vehicle off course, causing inaccurate execution. To reduce the errors caused by these unexpected events, the execution of each action is controlled by 'servoing' on prominent visual features in the environment. These features may be objects, such as prominent buildings, or they may be local features, such as easily identifiable corners, door frames, or baseboards. Servoing occurs on three nested levels:

'action-level servoing' is used to maintain the accuracy of each primitive action executed by the vehicle but does not relate the vehicle's position to its progress towards the goal;

¹This research has been supported in part by the Defense Advanced Research Projects Agency under RADC contract F30602-87-C-0140 and Army ETL contracts DACA76-89-C-0016 and DACA76-89-C-0017.

'plan-level servoing' uses the milestones defined in the plan to relate the current location of the robot to the plan and environmental model;

'goal-level servoing' attempts to relocate the robot when it becomes lost; this level of servoing is not discussed here.

1.1.2.1 Action-Level Servoing

The Denning Mobile Robotics vehicle used in these experiments can execute two 'primitive' actions directly: TURN θ and a straight line MOVE d . Neither of these actions can be reliably executed in 'open loop' mode. The straight line MOVE action, for example, results in a curved path when executed and the vehicle can be significantly off the intended straight line trajectory at the end of the action. In actual experiments, executing a MOVE 40' has resulted in the vehicle being as much as a foot off the intended straight line after 20', with the error increasing.

In action-level servoing, the primitive action MOVE 40', for example, is broken up into a sequence of smaller moves, say MOVE 2'. The 2D appearance information contained in the environmental model is used to generate two dimensional correlation templates for prominent visual features. From the predicted location of these features in the image, a search window is established and the templates are correlated with the image to establish their image location. Using the measured discrepancy between predicted and actual locations, the heading of the vehicle is modified to reduce the error and the next sub-action is executed. The process is repeated until the primitive action is complete. In actual experiments, the use of action-level servoing has maintained the vehicle within 1/4" of the intended straight line motion over a 40' move. Details on action-level servoing and more complete experimental results may be found in (Fennema and Hanson 1990a).

1.1.2.2 Plan-Level Servoing

Plan-level servoing is designed to ensure proper execution of a plan step prior to initiating the next step by relating the progress of the vehicle towards the goal to the environmental model. This is accomplished by matching the milestones defined in the plan sketch to the image (2D matching) followed by a 3D pose refinement step to determine the relationship between the vehicle and the environmental frame of reference.

Two different approaches to 2D model matching have been developed. The most recent approach has been to use the same feature extraction and 2D correlation methods used in action-level servoing. Since the vehicle has been tracking these points during action-level servoing, it is unlikely that there will be a large discrepancy between where the vehicle believes it is and where it actually is. Consequently, plan-level servoing involves only the additional step of 3D pose

refinement, using the matched points, in order to recover its position. In an initial set of indoor experiments described in (Fennema and Hanson 1990a), the vehicle was able to recover its position to within a quarter of an inch after being displaced up to 6 inches from where it thought it was, using landmarks approximately 30 feet away. Additional experiments with this technique are planned for the near future.

An earlier approach (Beveridge, Weiss et al. 1989; Beveridge, Weiss et al. 1990) matched straight lines extracted from the image to model lines projected to the image plane using the assumed location of the vehicle. During this past year, the two-dimensional matching scheme has been extended to include determination of scale, as well as the rotation and translation parameters yielding the best fit. The model-to-image line correspondences determined during 2D matching are used as the input to the 3D pose computation step. The emphasis over the past year has been on improving both the reliability and efficiency of the search processes.

The 3D pose refinement technique developed earlier works with either points or lines (Kumar 1989; Kumar and Hanson 1990a; Kumar and Hanson 1990b; Kumar and Hanson 1989a) and has been extended to be robust in the presence of outliers. The robustness is achieved at a computational cost, since the median of the error function is minimized by combinatorial methods over the subset space of all matched image and model lines. However, the method is capable of handling up to 49.9% outliers. In a recent paper (Kumar and Hanson 1990a), the superiority of the least-median squares algorithm over traditional least-mean squares algorithms as well as those based on statistical M-estimation techniques was established. The sensitivity of pose refinement and other related 3D inference methods to inaccurate estimates of the image center and focal length has been theoretically established and experimentally validated (Kumar and Hanson 1990b). The results show that for 'small' field of view imaging systems, incorrect knowledge of the camera center does not affect the recovered location of the camera significantly. The error in the recovered orientation of the camera is linearly related to the error in the estimate of the location of the center of the imaging system.

1.2 Automated Model Extension

The construction of positionally accurate environmental models is a time consuming, tedious task. Ultimately, the only feasible approach for vehicles which are required to interact with large scale changing environments is to provide them with methods for automatically acquiring their internal models during goal-oriented activities or unrestricted exploration.

Two preliminary experiments have been performed using the 3D pose refinement algorithm to extend a partial model from a set of known points to include unknown points;

these experiments are described in more detail in (Kumar and Hanson 1990b). The known model points are used to locate new points in the world coordinate system from pose refinement and triangulation over the induced stereo baseline obtained from a pair of 3D poses (e.g. location and orientation of the camera for each image). In both experiments, an image sequence was obtained for which the three-dimensional location of a set of points in the environment was known (the model). Image features are tracked over a sequence of frames using a token-based line tracker (Williams and Hanson 1988a; Williams and Hanson 1988b; Williams and Hanson 1988c), which provides the token correspondences. The 3D pose estimation algorithm described earlier is applied to each frame to map each feature into a stable world coordinate frame. The 3D pseudo-intersection of the rays passing through the camera center and the image feature point in each image frame is found using an optimization technique which minimizes the sum-of-squares of the perpendicular distances from the 3D pseudo-intersection point to the rays. In effect, this induces a stereo baseline between frames from which the 3D coordinates of the unknown features can be obtained by triangulation. Note that the computation of the location of new points in the world coordinate system is not sensitive to accurate estimation of the image center.

1.3 Automatic Acquisition of Environmental Models

The problem of acquiring models or modifying incorrect models is an important aspect of object recognition and navigation. The major functional requirements of modeling for these tasks are: accurate prediction of visual features, accurate surface orientation and curvature, and accurate feature dimensions.

We are currently building a system for acquisition of models from image data under known motion generated by a camera mounted on an arm in a robot workcell. In order to obtain accurate depth and curvature information, an extension of the Giblin and Weiss algorithm (Giblin and Weiss 1987) is being used. This algorithm computes depth and curvature by tracking contours in three successive images. The surfaces need not be smooth and the algorithm can use creases (tangent discontinuities) as well as extremal contours and surface markings. This produces 3-D contours and curvatures to which a surface can be fit.

There are many types of surface that one can fit to this 3-D data. We have chosen a representation based on a vertices, edges, and faces. This type of model is supported by Geometer (Connolly 1989; Connolly, Kapur et al. 1989) which provides an environment that includes both planar and algebraic faces.

1.4 Image-Based Navigation Using 360 Degree Views

Mobile robot navigation has proven to be a difficult task, and when a system must be capable of automatically acquiring 3D environmental models, it is currently beyond the state-of-the-art. We are developing a quite novel approach to robot navigation (Hong, Tan et al. 1990) that allows environmental information to be acquired in terms of a set of images of the world taken at a set of target locations. The robot navigates through the world by moving between neighboring target locations using an image-based local homing algorithm. Such an approach is feasible only because the system utilizes an unusual imaging system that provides 360 degree views of the scene in an extremely compact form.

The imaging system is comprised of a spherical mirror mounted above a video camera that is pointed upwards so that a 360° hemispherical view of the world is obtained as a circular extreme "fish-eye" image. This spherical imaging system so greatly distorts the scene during projection that the image changes dramatically as the robot moves, while maintaining visibility of the whole environment so that both objects that are in the path of movement as well as objects just passed will remain visible. There is, however, a projective invariant on the horizon line, or in this case the horizon circle. As the robot moves on a planar ground surface, distinctive world features (i.e. landmarks) that project to points on the horizon circle remain on the circle. In addition, each feature other than the points directly in front of and behind the robot slide around the horizon circle as a function of the robot movement and surface distance. A one-dimensional circular "location signature" is extracted from the hemispherical image by sampling along the horizon circle at angular intervals (in our experiments 1°), allowing any resolution image to be compressed into a 360-byte location signature.

Large scale navigation is then decomposed into a sequence of small-scale navigation tasks by local homing. Around each target location, there is a "capture radius" that allows comparison of landmarks in the current and target location signatures to determine a motion to reduce the difference and thereby home in on the local target in a series of small steps. Thus, a compact 360° representation of the environment and an image-based qualitative homing algorithm allows a mobile robot to navigate without explicitly inferring three-dimensional structure from the image. Experiments in typical indoor rooms and corridors have been successful along paths that involve as many as 17 target locations for incremental homing. This research is an ongoing effort and the feasibility of sampling two-dimensional space for general goal-oriented navigation is being examined.

2. Motion and Stereo Processing

2.1 A Framework for the Integrated Processing of Stereo and Motion

Work is in progress on understanding the *dynamics* of a scene as viewed by a stereo pair of cameras undergoing arbitrary motion. This subsumes both the analysis of static stereo imagery at one time instant to obtain the static disparity between the two images and thereby depth, and the analysis of a monocular motion pair to obtain the optic flow for a pair of frames and thereby relative motion and depth. Thus, we are specifically interested in a reconstruction paradigm which can be categorized as *binocular motion*, in order to obtain additional constraints on the recovery of motion and depth without depending on one unique (and possibly erroneous) source for the depth.

A promising approach utilizes the ratio of the relative flow between the image pairs to the disparity as a function of the motion in depth parameters (Balasubramanyam and Snyder 1988; Waxman and Duncan 1986). The vectors parallel to the real instantaneous 3D velocity scaled by the depth of the point, located at the image of the 3D point, can be extracted using purely image measurable quantities. This field of scaled 3D vectors is called the *p-field*. The *p-field* is interesting from the point of view of binocular motion since it implies that at the image level, where normally only 2D entities were available, it is now possible to examine and exploit the nature of 3D phenomena directly.

We are currently examining the use of the *p-field* as a framework within which to represent both the problem of occlusion and discontinuity (Balasubramanyam and Weiss 1989) detection and flow/disparity computation as well as computation of the 3D motion itself. For instance, observing that the *p-vector* is a scaled version of the real 3D motion vector, it seems more appropriate to impose smoothness on this vector since this is closer to the assumption of smooth 3D motion, rather than on flow smoothness. This was briefly examined in (Scott 1986) but not within the framework of binocular motion.

It may be possible to use the *p-field* for the interpretation of available flow and disparity information for the estimation of the motion parameters. For instance, in the case of ideal pure translation, the *p-field* directly yields the direction of translation. In the case of general motion, we are examining several possible algorithms for the computation of the motion parameters.

2.2 Smoothness Constraints For Optical Flow & Surface Interpolation

Gradient-based approaches to the computation of optical flow often use a minimization technique incorporating a smoothness constraint on the optical flow field.

Smoothness constraints are also of interest in surface interpolation, where they are known as "performance functions." All known smoothness constraints used to compute optical flow have a subtle property, namely that they do not mix derivatives of different components of the optical flow field.

Snyder (Snyder 1990) presents an analysis of smoothness constraints which do not satisfy this 'decoupled' property, but rather in which derivatives of different components of the flow can interact. By using representation theory of the group of Euclidean motions in the image plane, he uses the single assumption that the smoothness constraint is invariant under this group of transformations to generate a *complete* list of all possible invariant smoothness constraints.

The constraints are represented as type (p,q) , by which it is meant that they are quadratic in p^{th} derivatives of the optical flow field, and in q^{th} derivatives of the grey level image intensity function. This is done explicitly for the values $0 \leq p,q \leq 2$. It appears that of these smoothness constraints, excepting those linear combinations which are decoupled, are new. In addition, he finds all invariant "performance measures" used in surface interpolation, when the performance measure is quadratic in no higher than fourth derivatives of the object function.

2.3 Orientation Statistics for Modeling 3D Lines and Planes

One useful technique for deriving orientation information from static images is the estimation of a unit vector perpendicular to a number of derived unit vectors. For instance, under perspective projection a ray pointing towards the intersection of a group of converging image line segments is perpendicular to their projection plane normals. This has applications in finding vanishing points and in locating the focus of expansion of a pure translational flow field. Furthermore, the normal to a planar surface is perpendicular to the direction of all lines lying on that surface.

The problem of estimating a vector mutually perpendicular to several unit vectors can be characterized as estimating the polar axis of a great circle on the unit sphere. Bingham's distribution, which represents the intersection of a 3D Gaussian distribution with the surface of the unit sphere, is introduced to describe both equatorial and bipolar distributions of unit vectors. Statistical parameter estimation based on Bingham's distribution can be used to solve for the polar axis of a great circle of points and to represent the statistical uncertainty in the orientation estimate, but the procedure is somewhat expensive computationally. Collins and Weiss (Collins and Weiss 1990) develop a more convenient alternative based on linear-least-squares plane fitting. In addition, they consider the problem of estimating the orientation and uncertainty of the

cross product of two uncertain unit vectors. The tentative solution is to form a Gaussian approximation to the "intersection" of two equatorial Bingham distributions.

The above methods are illustrated using two examples (Collins and Weiss 1990). The first involves reconstruction of planar surfaces using stereo line correspondences. If relative pose of the stereo cameras can be described as a pure translation, then the orientation of lines in the world can be computed as the cross product of the projection plane normals of its two corresponding images, one in each image plane. Given a set of lines hypothesized to lie on a single planar surface, the plane orientation and uncertainty can be computed as the pole of a great circle formed from the uncertain line orientation estimates.

The second example involves the analysis of vanishing points (Collins and Weiss 1989). The images of parallel 3D lines converge to a vanishing point in the projective image plane. A ray constructed from the camera focal point towards the vanishing point has the same 3D orientation as the original world lines. The line orientation and its approximate confidence region on the unit sphere is estimated as the polar axis of a great circle of projection plane normals. Furthermore, surface plane orientations are hypothesized as the cross product of these uncertain line directions.

2.4 Analysis of the Limits of Robustness of Correspondence-Based Structure from Motion

In spite of extensive research in correspondence-based motion analysis, a comprehensive algorithm-independent study of the theoretical limits on the accuracy of the computation of environmental depth is not available. In response to this situation, Dutta and Snyder have been examining (Dutta and Snyder 1990) the robustness of correspondence-based approaches to structure from motion.

Their analysis shows, in an *algorithm-independent* way, that small absolute errors in image displacements cause absolute errors in rotational motion parameters significant enough to lead to large relative errors in the determination of environmental depth. Even if the motion parameters are known almost exactly, such as by sophisticated navigation systems, small errors in image displacements still lead to large errors in depth for environmental points whose distance from the camera is greater than a few multiples of the total translation in depth of the camera.

2.5 Comparative Results of Four Motion Algorithms

In an earlier paper, Sawhney (Sawhney and Oliensis 1989) presented a new technique for recovering motion and structure through image trajectories of rotational motion. A closed form solution was presented for the problem of recovering the 3D circular trajectory of a point given its

conic trajectory in the image plane. It was also demonstrated that when small sections of the conic arc in the image are used as the input for a trajectory description, one obtains very unreliable estimates of the underlying trajectory. Hand-grouped sets of trajectories were used and it was conjectured that if spatio-temporal data from proximal points could be grouped and trajectories fit to the grouped data, reliable combined trajectory descriptions and accurate 3D results could be obtained.

An algorithm has been developed which uses commonality of motion to first incrementally group point tracks and then fits conic sections to a subset of these using an optimization technique over a joint error measure. The error measure uses a parameterization which makes the common and independent parameters of each trajectory explicit. The closed form solution was presented in (Sawhney and Oliensis 1989).

The algorithm has been applied to several image sequences; the results are summarized in (Sawhney and Hanson 1990; Sawhney and Oliensis 1990). In addition, the results have been compared with two other motion algorithms: Adiv (Adiv 1985), Horn's relative orientation algorithm (Horn 1988), as well as Kumar's pose refinement algorithm (Kumar and Hanson 1989a; Kumar and Hanson 1989b). The results are preliminary and represent a continuing effort in understanding robust 3D reconstruction from monocular motion. More accurate results applied to a more varied data set awaits precise calibration of our cameras.

3. Interpretation of Static Scenes

3.1 Learning 3D Object Recognition Strategies

A general system for object and scene interpretation, called the Schema System, has evolved as part of a long-term research effort at UMass (Draper 1989; Draper, Brolio et al. 1989; Hanson and Riseman 1978; Hanson and Riseman 1987; Riseman and Hanson 1984). The results of successful experiments in the outdoor scene domain has led to the not surprising conclusion that a declarative representation of knowledge would be more useful for future work, and in particular, automatic mechanisms for learning object recognition strategies (Hanson and Riseman 1989).

The basis for recognizing objects in complex outdoor scenes varies widely in terms of the processes utilized, the reliability of the information extracted, the efficiency of the underlying mechanisms, and the manner in which the evidence is combined into an object hypothesis. All of this information is certainly object- and domain-dependent. Some objects can be distinguished on the basis of color, while others can only be identified by scene and object context. Three-dimensional information about shape or texture of some objects might be recovered through bottom-

up vanishing point analysis, while the locations of other objects are more easily determined by model-based point matching.

The problem of learning how to recognize an object is being addressed in (Draper and Riseman 1990). The system is given a description of the object and a set of user-interpreted training images. The task is to build the most efficient object recognition strategy possible within performance constraints set by the user. Three-dimensional 3D object recognition is approached within a generate-and-verify paradigm. The task of learning to generate the minimal necessary set of hypotheses is phrased as a search problem. The task of learning to verify a hypothesis is cast as a classification problem, followed by graph optimization.

3.2 Perceptual Organization of Occluding Contours

Contours corresponding to surface boundaries are readily perceived or completed by human observers even when local evidence in the form of measurable image brightness gradients is completely absent. A classic example of the former is the Kanizsa triangle, in which the illusory contours of the 'occluding' triangle are visually compelling, even though there is scant evidence for their existence. An example of completion occurs when one surface is partially occluded by a second (opaque) surface.

Williams (Williams 1990) has developed a system for perceptually organizing surface boundaries based on figural clues alone, although results have only been demonstrated in the 'Colorforms' domain and other simple scenes. The system has, however, successfully extracted Kanizsa's occluding triangle and has correctly analyzed relatively complex scenes containing multiple occluding surfaces. Detailed results are presented in Williams (Williams 1990). The current system is designed to complete gaps in the straight sections of occluded contours but isn't yet able to cope with more complex occlusions, such as missing corners or missing sides.

In Williams' system, the mechanisms of occlusion of one surface by another are captured in a set of integer linear constraints. These constraints ensure that the outputs of a contour grouping process is physically valid and consistent with the image evidence. Among the many feasible solutions, the most compelling is the solution which best explains the presence and form of the image structure. The problem of computing a complete and consistent surface boundary representation is thus reduced to solving an integer linear program.

3.3 Perceptual Organization of Curves

Dolan (Dolan and Weiss 1989) is extending the perceptual grouping mechanisms developed by Boldt (Boldt, Weiss et al. 1989) for straight lines to the case of general curves. Like the straight line system, the curve grouping algorithm

relies on the Gestalt principles of proximity and good continuation and employs an iterative token-based approach to search for and describe significant curve structures (including straight lines, conic arcs, inflections, corners, and cusps).

The system iterates a cycle of linking, grouping, and replacement over a range of perceptual scales, but within each iteration processing occurs independently at each token. Each token is linked to other tokens that are likely to be its neighbors along some contour. Sequences of linked tokens are analyzed and classified based on the geometric structure they exhibit. Appropriate replacement tokens are then generated to explicitly describe and replace each sequence. Beginning with initial edge tokens (unit tangents centered at edge locations), curved structure is discovered in a bottom-up, local-to-global fashion and a multi-scale description results. The computational complexity inherent in any grouping process is managed here by searching locally within a perceptual window (which defines the local scale) and by explicitly replacing a sequence of tokens by a single token at the next scale.

Since the work previously reported in (Dolan and Weiss 1989), a parallel version of the grouping algorithm has been implemented in anticipation of parallel hardware. Here, the grouping process is simultaneously applied to the perceptual window (i.e. context) around each token for potential grouping and replacement, and parallel replacement of the aggregate tokens is assumed to take place simultaneously. A consequence of a highly distributed and parallel grouping process is that redundant descriptions arise because the contexts of nearby tokens overlap, and overlapping aggregate tokens are produced. Dolan is currently developing methods to identify and eliminate such redundancies by representing multiple types of relationships in the link graph; this will allow redundancy, as well textural structures to be dealt with in this parallel framework.

3.4 View Variation of Line Segment Features

Model-directed object recognition becomes much more difficult when the viewpoint of the three-dimensional object is unknown. A popular approach is based upon the use of multiple two-dimensional views of three-dimensional structures, and is referred to under a variety of terms such as "aspect graphs", "generic views", and "characteristic views" (Burns 1987; Burns and Kitchen 1987a; Burns and Kitchen 1987b; Burns and Kitchen 1988; Ikeuchi 1987). If such systems are going to be effective, a clear understanding is required of the manner in which the features of 2D projections vary as a function of the 3D viewing position of the object. It is important to find metric features of an object whose variation is small over a large range of views in order to constrain the number that must be stored.

Burns (Burns, Weiss et al. 1990) presents an analysis of the variation of point-set and line-segment features with respect to view. Although there are well known special-case invariants for four points, such as the cross ratio, there is no scalar invariant for an arbitrary number of points in general position, whether one uses true perspective, weak perspective or orthographic projection. The paper focuses on variation of features with respect to views of line segment pairs under weak perspective, a commonly used projection model in 3D recognition. The variation is a function of both the particular feature and the particular configuration of 3D line segments. The features studied are: the relative orientation, size and position of one line segment with respect to another, and the affine coordinates of one endpoint in terms of the other three.

The information in the view-variation analysis allows determination of semi-invariant features of an object over areas of the 3D viewing sphere, i.e. features which have a small variation over a large fraction of views. The relationships between the range of feature variation and the fraction of views are presented in a series of graphs for the features described above, and for varying instances of 3D line segments pairs. The mathematical analysis embodied in this paper is generally relevant to techniques for matching 3D models to 2D images.

3.5 Recovering Shape from Shading

Shape from shading has traditionally been considered an ill-posed problem. However, in recent work, Oliensis (Oliensis 1990a; Oliensis 1990c) has demonstrated that the solutions to shape from shading are often well-determined, with little or no ambiguity. For the case of illumination that is symmetric around the viewing direction (i.e. the light source is behind the camera), it was shown in (Oliensis 1989) that there is in general a *unique* solution to shape from shading. This proof is valid for *general* Lambertian objects (without holes), and is the first proof that the problem of shape from shading can be well-posed in general. These arguments were extended to the case of general illumination direction in (Oliensis 1990c), where it was demonstrated that, in this case also, the solutions to shape from shading are strongly constrained over much of the image. These results follow from a combination of local uniqueness theorems and global arguments concerning the properties of the flow of characteristic strips, both derived from the mathematical theory of dynamical systems theory. The essential constraints restricting the solution space are shown to be provided by the *singular points* in the image. Also, characteristic strips are given a simple interpretation as space curves, and demonstrated to be *independent of the viewing direction*.

It has long been an open question whether the image of the occluding boundary provides additional constraints on the solution to shape from shading. In (Oliensis 1990c), it is proven analytically that the answer to this question is

negative. Specifically, for a local image patch containing a portion of the boundary, the problem of shape reconstruction is shown to be ill-posed. Shape reconstruction is actually more ambiguous in the neighborhood of an occluding boundary segment than it is in the neighborhood of an interior image curve. The proof, which applies to a Lambertian surface illuminated from a general light source direction, is based on recasting the basic characteristic strip equations of Horn in a form that is completely *non-singular* on the occluding boundary.

Also, an example is presented in (Oliensis 1990c) in which a small image region bordering the image of the occluding boundary yields an ambiguous shape reconstruction, even though the image contains both singular points and the whole of the occluding boundary. This example demonstrates that shape from shading can be well-posed *and* ill-posed simultaneously: although the shape corresponding to most of the image is actually uniquely determined, the shape corresponding to the specified small image region is ill-determined. It is argued that, in general, these 'ill-posed' regions are probably small fractions of the image, but that they can occur frequently, in images both with and without visible occluding boundaries, and in practice may lead to instabilities and errors in shape reconstruction algorithms.

Finally, Oliensis has developed (Oliensis 1990b; Oliensis 1990c) a new local algorithm for reconstructing shape from shading using a general quadratic surface model. The new constraints for shape from shading should be investigated for their potential for robust surface reconstruction.

4. The Image Understanding Architecture

The Image Understanding Architecture (IUA) consists of three levels of tightly coupled array of parallel processors (Weems, Levitan et al. 1989). Work on the IUA has advanced in four areas in the preceding year through cooperative efforts by Hughes Research Labs and the University of Massachusetts. A generalized routing algorithm for the low-level processor has been developed, an Apply compiler for the low level has been implemented, the IUA simulator and tools have been enhanced, and assembly of the prototype system has begun. We have also started development of a data parallel C for the low level, continued planning of the next generation of the DARPA IU Benchmark, and started the development of the second generation IUA and the design of the third generation.

4.1 Routing On The CAAPP

The low-level processor of the IUA is a square mesh of processing elements, augmented with a second (reconfigurable) mesh, called the Coterie Network (Weems and Rana 1990). Normally, a mesh network is considered to be ill-suited for permutation routing because of the square-root of N diameter of the network. Other architectures, such

as the Connection Machine and Masspar have devoted a significant amount of additional hardware to support fast permutation routing. These additions take the form of hypercube or crossbar networks with sophisticated controllers. On the other hand, the Coterie Network merely adds a set of switches and some pre-charge logic to each processor, with no increase in the number of physical connections between processors.

Using the Coterie Network and the fast summary capabilities (Rana and Weems 1990) of the CAAPP, we have developed an adaptive routing algorithm that is at worst an order of magnitude slower than routing on the Connection Machine, and in many cases is up to two orders of magnitude faster. The algorithm is actually a collection of different algorithms, each suited to optimizing performance on different types of permutation. The first step is to quickly identify some gross features of the data to be routed, such as the density of messages and the average distance that they will travel, and then select the appropriate algorithm.

Most of the routing algorithms are straightforward and will not be repeated here. The reader is referred to (Herbordt and Weems 1990a; Herbordt and Weems 1990b; Herbordt, Weems et al. 1990) for the details. However, one algorithm is particularly novel, as it uses the Coterie Network to route data in a manner similar to the MIMD wormhole routing technique (Dally and Seitz 1987) on the SIMD CAAPP.

Each message has a header that knows its destination. Assuming that there is no blocking, a message header enters the network along a row, with message packets following it like a train of railroad cars behind an engine. When the header reaches the column of its destination address, it switches direction and the packets follow it along the column. The message is then consumed by the destination processor.

If the path of the header is blocked at any point by another message, the header must stop and wait for a clear path. In addition, the trailing packets must also wait. The problem with this approach in a normal SIMD system is that the header must notify each processor containing a trailing packet. Such notification takes as many steps as there are packets in the longest blocked message. When the path clears, the notification must be repeated so that the train can start to move again.

However, with the Coterie Network we can dynamically form a bus that maps onto a train of packets. Each train is connected to an independent bus with the header designated the bus master. When the header is blocked it merely sends a one-bit message out on the bus, and every trailing packet is notified in a single cycle. When the blockage clears, one more cycle is required to tell the trailing packets to advance.

4.2 Apply Compiler for the CAAPP

We have implemented a version of the Apply language (Hamey, Webb et al. 1987) for the low-level processor of the IUA. The compiler generates C code that can be integrated with the C that is currently used to program the CAAPP. It permits us to rapidly write functions for the CAAPP that perform local window based operations on images. We have also compiled most of the Webb library, supplied with Apply, for the CAAPP and tested their performance. The details of this effort can be found in (Scudder and Weems 1990).

4.3 Enhancements to IUA Simulator and Tools

The IUA simulator includes an elegant user interface that provides extensive interaction with the processors for debugging software (Weems and Burrill 1990). However, the interface has only been available on Sun workstations because it was written with the SunViews windowing system. The simulator has now been converted to run with X-Windows, which will greatly enhance its portability. In addition, the use of X allows a user to run a simulation on a powerful server while displaying the system status on a low-cost workstation or X-terminal.

The low level of the IUA was originally programmed using a FORTH interpreter because the simplicity and extensibility of FORTH allowed us to easily add data parallel constructs and run with low interpretation overhead. While suitable for developing simple applications, FORTH is severely limiting as applications grow in size. We have thus implemented a C preprocessor that allows programming of the CAAPP in a manner similar to the C-PARIS facility on the Connection Machine. That is, programs are developed using C control structures, but CAAPP operations are executed through calls to an extensive library of subroutines.

4.4 Prototype System Assembly

All of the custom chips used in the IUA have been assembled and tested at the Hughes Research Labs. A breadboard has been built that exercises CAAPP chips and ICAP chips running together and communicating with each other. The backplane for the prototype has been assembled and tested. In addition, the first processor board, containing 256 CAAPP processors and four ICAP processors has been assembled and is being tested prior to fabrication of the remaining 15 boards that will make up the prototype. Completion of the prototype and delivery to Umass is expected by the end of Summer 1990.

4.5 Current Work

We are now working on a data parallel extension to C that will explicitly support an image plane data type on the CAAPP level of the IUA. This language will permit the user to define image planes of different sizes, automatically mapping them onto virtual processors. We are also looking

into creating a C* compiler for the CAAPP, in order to provide portability of code with the Connection Machine and Masspar.

As recommended by the DARPA IU Benchmark Workshop participants, much of the benchmark (Weems, Riseman et al. 1990; Weems, Riseman et al. 1988) has been recoded as a set of library routines which are called by the core of the benchmark. The most complex portion of the benchmark, the graph matcher, must still be recoded to achieve greater generality. We are also starting to plan the second level benchmark, which will incorporate tracking of moving objects over a sequence of images.

The development of the second generation of the IUA has already begun (Weems, Hanson et al. 1990). The only significant changes to the architecture are that 256 CAAPP processors will be placed on each custom chip, allowing 4K processors to be placed on a single board; and the ICAP processors will be upgraded to the TMS320C30 processor. The latter change makes each ICAP processor a 32-bit, 32 MFLOP device instead of the 16-bit, 5 MIPS devices that are currently used. With the TMS320C30 we will also be able to expand the memory at each node, and also support a multi-tasking kernel.

The third generation of the IUA is already being designed. While the second generation is mainly a technology enhancement of the first generation, the third generation will be substantially different in its architecture. The low level will change from bit-serial processors to 8-bit processors, each with a much larger on-chip cache, and additional support for floating point. The communication bandwidth at the intermediate level will be greatly enhanced (Rana, Weems et al. 1988). At the high level, we finally expect systems to become large enough that a RISC-based multiprocessor will be employed. Currently the prototype is small enough that only a single processor has been incorporated at the high level. A full-scale implementation of the third generation (256K CAAPP PE's, 1K ICAP processors, 64 SPA processors) would achieve roughly a terra-op in performance on 32-bit integer arithmetic.

5. References

- Adiv, G. (1985). "Determining 3-D Motion and Structure from Optical Flow Generated by Several Moving Objects," IEEE PAMI, Vol. 7(4), pp. 384-401.
- Balasubramanyam, P. and M. Snyder. (1988). "Computation of Motion-in Depth Parameters: A First Step in Stereoscopic Motion Interpretation," Proc. of DARPA Image Understanding Workshop, Cambridge, MA, pp. 907-920.
- Balasubramanyam, P. and R. Weiss. (1989). "Early Identification of Occlusion in Stereo-Motion Image Sequences," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA, pp. 1032-1037.
- Beveridge, J. R., R. Weiss and E. Riseman. (1989). "Optimization of 2-Dimensional Model Matching," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA, pp. 815-830.
- Beveridge, J. R., R. Weiss and E. M. Riseman. (1990). "Combinatorial Optimization Applied to Variable Scale 2D model Matching," Proc. of IEEE Intl. Conf. on Pattern Recognition, Atlantic City, NJ, pp. 18-23.
- Boldt, M., R. Weiss and E. Riseman. (1989). "Token-Based Extraction of Straight Lines," IEEE-SMC.(19)6, December 1989, pp.1581-1594.
- Burns, J. B. (1987). "Recognition in 2D Images of 3D Objects from Large Model Bases Using Prediction Hierarchies," Proc. of International Joint Conference on Artificial Intelligence, Milan, pp. 763-766.
- Burns, J. B. and L. J. Kitchen. (1987a). "An Approach to Recognition in 2D Images of 3D Objects from Large Model Bases," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR 87-85.
- Burns, J. B. and L. J. Kitchen. (1987b). "Rapid Recognition out of Large Model Bases Using Prediction Hierarchies and Machine Parallelism," Proc. of SPIE Conf. on Intelligent Robots and Computer Vision.
- Burns, J. B. and L. J. Kitchen. (1988). "Rapid Object Recognition from a Large Model Base using Prediction Hierarchies," Proc. of DARPA Image Understanding Workshop, Cambridge, MA, pp. 711-719.
- Burns, J. B., R. Weiss and E. M. Riseman. (1990). "View Variation of Point Set and Line Segment Features," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.
- Collins, R. T. and R. Weiss. (1989). "An efficient and Accurate Method for Computing Vanishing Points," Proc. SPIE Workshop on Image Understanding and Machine Vision, Vol. 14, Optical Society of America, Washington, DC, pp.92-94.
- Collins, R. T. and R. Weiss. (1990). "Orientation Statistics for Modeling 3D Lines and Planes," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.
- Connolly, C. (1989). "Geometer: Solid Modelling and Algebraic Manipulation," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR In Preparation.

- Connolly, C., D. Kapur, J. Mundy and R. Weiss. (1989). "Geometer: A System for Modelling and Algebraic Manipulation," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA, pp. 797-804.
- Dally, W. J. and C. L. Seitz. (1987). "Deadlock Free Routing in Multiprocessor Interconnection Networks," IEEE Trans. on Computers, Vol. C-36(5).
- Dolan, J. and R. Weiss. (1989). "Perceptual Grouping of Curved Lines," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA, pp. 1135-1145.
- Draper, B. (1989). "Integrating Top-Down Control with Intermediate-Level Vision," Proc. of SPIE Conference on Applications of AI - VII, Orlando, FL.
- Draper, B., J. Brolio, R. Collins, A. Hanson and E. Riseman. (1989). "The Schema System," IJCV, Vol. 2(3), pp. 209-250.
- Draper, B. and E. M. Riseman. (1990). "Learning 3D Object Recognition Strategies," Dept. of Computer and Information Science, University of Massachusetts (Amherst).
- Dutta, R. and M. Snyder. (1990). "Robustness of Correspondence-Based Structure from Motion," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.
- Fennema, C. L. and A. R. Hanson. (1990a). "Experiments in Autonomous Navigation," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.
- Fennema, C. and A. R. Hanson. (1990b). "Towards Autonomous Mobile Robot Navigation," IEEE SMC.
- Giblin, P. and R. Weiss. (1987). "Reconstruction of Surfaces from Profiles," Proc. of First IEEE International Conference on Computer Vision, London.
- Hamey, L. G. C., J. A. Webb and I. C. Wu. (1987). "Low-Level Vision on Warp and the Apply Programming Model," Dept. of Robotics Institute, Carnegie Mellon University, TR CMU-RI-TR-87.
- Hanson, A. and E. Riseman. (1978). "VISIONS: A computer System for Interpreting Scenes" in Computer Vision Systems (A. Hanson and E. Riseman, Ed.), New York: Academic Press.
- Hanson, A. and E. Riseman. (1987). "The VISIONS Image Understanding System" in Advances in Computer Vision (C. Brown, Ed.), Hillsdale, NJ: Erlbaum Press.
- Hanson, A. and E. Riseman. (1989). "Progress in Computer Vision at the University of Massachusetts," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA, pp. 49-55.
- Herbordt, M. C. and C. C. Weems. (1990a). "General Routing on the Lowest Level of the Image Understanding Architecture," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.
- Herbordt, M. T. and C. C. Weems. (1990b). "Message Passing Algorithms for a SIMD Torus with Coterie," Proc. of ACM Intl. Symposium on Parallel Algorithms and Architectures, Crete.
- Herbordt, M. T., C. C. Weems and D. B. Shu. (1990). "Routing on the CAAPP," Proc. of IEEE Intl. Conf. on Pattern Recognition, Atlantic City, NJ, pp. 467-471.
- Hong, J., X. Tan, B. Pinette, R. Weiss and E. M. Riseman. (1990). "Image Based Navigation Using 360 Degree Views," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.
- Horn, B. K. P. (1988). "Relative Orientation," Proc. of DARPA Image Understanding Workshop, pp. 826-837.
- Ikeuchi, K. (1987). "Generating an Interpretation Tree from a CAD Model for 3D-Object Recognition in Bin-Picking Tasks," IJCV, Vol. 1(2), pp. 145-165.
- Kumar, R. (1989). "Determination of Camera Location and Orientation," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA, pp. 870-881.
- Kumar, R. and A. R. Hanson. (1990a). "Analysis of Different Robust Methods for Pose Refinement," Proc. of IEEE Workshop on Robust Methods in Computer Vision, Seattle.
- Kumar, R. and A. R. Hanson. (1990b). "Pose Refinement: Application to Model Extension and Sensitivity to Camera Parameters," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.
- Kumar, T. and A. Hanson. (1989a). "Robust Estimation of Camera Location and Orientation from Noisy Data Having Outliers," Proc. of IEEE Workshop on Interpretation of 3D Scenes, Austin, TX, pp. 52-60.
- Kumar, T. and A. Hanson. (1989b). "Robust Estimation of Camera Location and Orientation from Noisy Data with Outliers," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR 89-120.
- Oliensis, J. (1989). "Existence and Uniqueness in Shape from Shading," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR 89-109.

Oliensis, J. (1990a). "Existence and Uniqueness in Shape from Shading," Proc. of IEEE Intl. Conf. on Pattern Recognition, Atlantic City, NJ, pp. 341-345.

Oliensis, J. (1990b). "New Results in Shape from Shading," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.

Oliensis, J. (1990c). "Shape from Shading as a Partially Ill-Posed Problem," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR 90-50.

Rana, D. and C. C. Weems. (1990). "A Feedback Concentrator for the Image Understanding Architecture," Proc. of IEEE Intl. Conf. on Pattern Recognition, Atlantic City, NJ, pp. 540-544.

Rana, D., C. C. Weems and S. P. Levitan. (1988). "An Easily Reconfigurable Circuit Switched Connection Network," Proc. of IEEE Intl. Symposium on Circuits and Systems, pp. 247-250.

Riseman, E. M. and A. R. Hanson. (1984). "A Methodology for the Development of General Knowledge-Based Vision Systems," Proc. of IEEE Workshop on Computer Vision: Representation and Control, pp. 159-170.

Sawhney, H. and A. R. Hanson. (1990). "Comparative Results of Some Motion Algorithms on Real Image Sequences," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.

Sawhney, H. and J. Oliensis. (1989). "Description and Interpretation of Rotational Motion from Image Trajectories," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA, pp. 992-1003.

Sawhney, H. and J. Oliensis. (1990). "Image Description and 3D Interpretation from Image Trajectories Under Rotational Motion," Dept. of Computer and Information Science, University of Massachusetts (Amherst).

Scott, G. L. (1986). "Smoothing the Optic Flow under Perspective Projection," Proc. of CVPR, pp. 504-509.

Scudder, M. and C. C. Weems. (1990). "An Apply Compiler for the CAAPP," Dept. of Computer and Information Science, University of Massachusetts (Amherst).

Snyder, M. (1990). "On the Calculation of Rigid Motion Parameters from the Essential Matrix," Dept. of Computer and Information Science, University of Massachusetts.

Waxman, A. and J. Duncan. (1986). "Binocular Image Flow: Steps Towards Stereo-Motion Fusion," IEEE PAMI, Vol. 8, pp. 715-729.

Weems, C. C. and J. R. Burrill. (1990). "The Image Understanding Architecture and its Programming Environment" in Parallel Architectures and Algorithms for Image Understanding (V. K. Prassana-Kumar, Ed.), Orlando, FL: Academic Press.

Weems, C. C., A. R. Hanson, E. M. Riseman, D. Rana, D. B. Shu and J. G. Nash. (1990). "An Overview of Architecture Research for Image Understanding at the University of Massachusetts," Proc. of IEEE Intl. Conf. on Pattern Recognition, Atlantic City, NJ, pp. 379-384.

Weems, C., S. Levitan, A. Hanson, E. Riseman, J. Nash and D. Shu. (1989). "The Image Understanding Architecture," IJCV, Vol. 2(3), pp. 251-282.

Weems, C. C. and D. Rana. (1990). "Reconfiguration in the Low and Intermediate Levels of the Image Understanding Architecture" in Reconfigurable SIMD Parallel Processors (H. Li, Ed.), Englewood Cliffs, NJ: Prentice-Hall. Also COINS Technical Report TR 90-10, University of Massachusetts (Amherst).

Weems, C. C., E. M. Riseman, A. R. Hanson and A. Rosenfeld. (1990). "An Integrated Image Understanding Benchmark for Parallel Processors," J. Parallel and Distributed Computing.

Weems, C., R. Riseman, A. Hanson and A. Rosenfeld. (1988). "An Integrated Image Understanding Benchmark: Recognition of a 2 1/2 D "Mobile"," Proc. of IEEE Conf. on Computer Vision, Ann Arbor, MI.

Williams, L. R. (1990). "Perceptual Organization of Occluding Contours," Proc. of 1990 DARPA Image Understanding Workshop, Pittsburgh, PA.

Williams, L. R. and A. Hanson. (1988a). "Depth From Looming Structure," Proc. of DARPA Image Understanding Workshop, Cambridge, MA, pp. 1047-1051.

Williams, L. R. and A. Hanson. (1988b). "Translating Optical Flow into Token Matches," Proc. of DARPA Image Understanding Workshop, Cambridge, MA, pp. 970-980.

Williams, L. R. and A. R. Hanson. (1988c). "Translating Optical Flow into Token Matches and Depth from Looming," Proc. of 2nd International Conference on Computer Vision, Tarpon Springs, FL, pp. 441-448.

Image Understanding Research at SRI International

Martin A. Fischler and Robert C. Bolles

Artificial Intelligence Center
SRI International
333 Ravenswood Avenue, Menlo Park, CA 94025

Abstract

The image understanding program at SRI International is a broad effort spanning the entire range of machine vision research. In this report we describe our progress in two domains: the first is concerned with modeling the earth's surface from aerial imaging sensors; the second is concerned with visual interpretation for ground-level vision and land navigation. In particular, we describe progress in stereo compilation and automated terrain modeling from aerial imagery; in interactive scene modeling and scene generation; in automatic image segmentation and delineation of man-made objects; in detecting and tracking moving objects; and in using knowledge beyond shape and immediate appearance to recognize objects in natural scenes and other complex domains.

1 Introduction

The overall goal of Image Understanding research at SRI International is to obtain solutions to fundamental problems in computer vision that are necessary to allow machines to model, manipulate, and understand their environment from sensor-acquired data and stored knowledge.

In this report we describe progress in two domains, aerial and ground-based vision.¹ The first is concerned with modeling the earth's surface from photographs taken from aircraft and satellites; the second is concerned with modeling a natural environment in real time from data taken by a robotic device moving through, and interacting with, this environment.

In the discussion of the first domain we describe our progress in developing stereo techniques for building terrain models from aerial imagery; interactive techniques for building three-dimensional models of man-made and cultural objects, and a new automatic technique for segmenting aerial images into coherent regions and for detecting and delineating man-made objects.

¹Supported by the Defense Advanced Research Projects Agency, under contracts DACA76-85-C-0004, MDA903-86-C-0084, and 89F737300.

In the discussion of ground-based vision we describe progress in developing techniques for building object descriptions that evolve gradually over time as more data are obtained, motion analysis techniques for detecting and tracking moving objects in data taken by moving sensors, and a new method for using contextual information to recognize natural objects, such as trees and bushes, in outdoor scenes.

An important theme in much of our current work is an emphasis on computational performance — especially through the development of algorithms capable of exploiting the new parallel machine architectures now available (e.g., the Connection Machinetm).²

2 Automated Terrain Modeling from Aerial Imagery

Stereo reconstruction is a critical task in cartography that has received a great deal of attention in the image understanding community ([Barnard90], [Barnard & Fischler], *these proceedings*). Its importance goes beyond the obvious application to constructing geometric models: understanding scene geometry is necessary for effective feature extraction and other scene analysis tasks. While considerable success has been achieved in important parts of the problem, there is no complete stereo-mapping system that can perform reliably in a wide variety of scene domains.

Historically, the computational modeling of stereo vision has been driven by a number of diverse motivations. The practical applications of automated stereo are so important, especially in cartography and robotics, that many engineering-oriented approaches have been tried. These often use "correlation" techniques: patches of intensities in one image are searched for in the other image by maximizing a measure of correlation or minimizing a measure of error. The other motivations, such as the desire to model biological stereo, involve a variety of techniques. Some are feature-based. discrete local features

²Use of a Connection Machine was provided by DARPA.

(usually edges) are matched across images, others use an approach in which a dense disparity map is the state variable of a system, stereo matching is then formulated as an optimization problem. find the best disparity map by maximizing an objective function that measures the "quality" of the map.

Our research strategy in this task is to develop new techniques for the key steps in the stereo process, such as matching and interpolation, and, in parallel, to integrate these new ideas with existing techniques in the context of an operational system. As part of this process SRI has implemented [Hannah85] and evaluated [Hannah88, Hannah89] a complete high-performance stereo system, STEREO SYS, that uses a combination of the correlation and feature-matching approaches. In a test of existing stereo systems on 12 pairs of digital images, conducted by the International Society of Photogrammetry, STEREO SYS was able to successfully process more of the images than any other system (11 out of the 12 pairs); while no formal ranking of the test results will be published, it appears that this system placed first (or very near the top) in the competition.

Another system we have developed for stereo matching is CYCLOPS [Barnard90]. This work began as a stochastic-optimization approach to stereo matching, but has recently evolved into a more complete system for cartographic terrain modeling, including software modules for camera modeling, epipolar resampling, the generation of regular-grid elevation maps, ortho-images, contour plots, and synthetic perspective views, in addition to the central task of image matching. One of the goals of this work has been to develop efficient stereo-processing methods for massively parallel SIMD architectures. The CYCLOPS system is implemented on the Connection Machine. The current implementation is capable of producing a dense terrain model (depth for every pixel) for a typical pair of 1024x1024 aerial stereo image in about eight minutes, using a Connection Machine with 4096 processors. First, camera model information is used to produce corrected images with only horizontal parallax. The corrected images are then matched with a multigrid optimization algorithm. Essentially, the matching algorithm is a stochastic regularization method that tries to find the flattest dense disparity map that matches the photometry with least error. It does so by iterating a microcanonical version of simulated annealing across several levels of a resolution pyramid, using the results from the coarser levels to initialize the optimization search at the finer levels. After the corrected images are matched the disparity measurements are converted into a dense but irregular mesh of depth measurements, which is then resampled into a grid of elevations with respect to regularly spaced ground coordinates.

One significant new development in the basic matching algorithm of CYCLOPS is the *three-pools mechanism*. Studies of anomalous stereo vision suggest that there

are three psychophysically distinct "pools" of disparity detectors, corresponding roughly to crossed (positive), uncrossed (negative), and near-zero disparity relative to the vergence point. CYCLOPS uses the same representation, coupled with several octaves of resolution hierarchy, to achieve efficiency without sacrificing dynamic range. At every level in the hierarchy, components of incremental disparity can assume only three local values: 1 (crossed), -1 (uncrossed), and zero. The base disparity, however, can grow by a factor of two across every level. The search space in any one level is therefore kept to a minimum, while the final composite disparity is allowed to have a substantial range. In addition to increasing efficiency, the three-pools mechanism leads to generally better (i.e., more nearly optimal) results.

While the stereo problem remains a focus of one segment of our research program, additional effort is now being devoted to developing an understanding of how knowledge of scene depth can be effectively used in the scene-partitioning and object-recognition tasks. The development of much faster hardware will open the door to a new stereo application: robot vision. Stereo obviously has a significant role to play in robotics, whether as a straightforward mensuration tool for industrial applications or as part of an integrated perceptual system of an autonomous, mobile robot.

3 Interactive Techniques for Scene Modeling: A Cartographic Modeling Environment

Manual photointerpretation is a difficult and time-consuming step in the compilation of cartographic information. However, fully automated techniques for this purpose are currently incapable of matching the human's ability to employ background knowledge, common sense, and reasoning in the image-interpretation task. Near-term solutions to computer-based cartography must include both interactive extraction techniques and new ways of using computer technology to provide the end-user with useful information in the form of both image and map-like interactive computer displays.

In order to support research in semiautomated and automated computer-based cartography, we have developed the SRI Cartographic Modeling Environment. In the context of an interactive workstation-based system, the user can manipulate multiple images; camera models, digital terrain elevation data, point, line, and area cartographic features, and a wide assortment of three-dimensional objects. Interactive capabilities include free-hand feature entry, feature editing in the context of task-based constraints, and adjustment of the scene viewpoint. Synthetic views of a scene from arbi-

rary viewpoints may be constructed using terrain and feature models in combination with texture maps acquired from aerial imagery. This ability to provide an end-user with an interactively controlled scene-viewing capability could eliminate the need to produce hard-copy maps in many application contexts. Additional applications include high-resolution cartographic compilation, direct utilization of cartographic products in digital form, and generation of mission-planning and training scenarios.

Recent work has focused on porting the CME to a UNIX/C platform (from its current LISP-machine implementation) in order to support technology transfer goals. Other work involves developing more flexible object representations, irregular terrain grids, and improved interfaces to other systems such as the SRI-developed Core Knowledge System. One especially important technical improvement involves sensor geometry extensions.

The SRI Cartographic Modeling Environment uses sensor geometry models in two principal ways: 1) projecting the 3D world coordinates into 2D sensor (pixel) coordinates, and 2) computing the intersection of a 3D ray (corresponding to a sensor pixel) with a terrain model. The basic CME system currently supports only central (perspective) projection and orthographic projection. In central projection, each point in 3-space is projected onto the camera sensor plane along a ray passing through a common point, the projection center. We are currently implementing a generic capability for dealing with non-central-projection sensor geometries. When accomplished, essential operations now supported for central projection imagery would also be supported for other types of (orbiting) sensors. These operations include:

1. Display of three-dimensional feature models that are cartographically registered to non-central-projection imagery.
2. Terrain rendering, using data acquired with any sensor geometry, to a format simulating any other sensor geometry. An example would be mapping non-central-projection imagery onto a terrain model and generating a simulated image showing the result viewed with a central-projection sensor.

In addition to implementing non-central-projection sensor geometries, we are also working on generating the expected flow field observed by a moving sensor. This work is an important component in the development of automatic techniques for improving terrain models from sequences of real imagery. By comparing the measured image flow in the real data with the predicted flow we can discover discrepancies between the terrain model and the actual terrain. Our goal is to use the depth-from-motion technologies developed at SRI to convert the discrepancies into improved depth estimates and thus a refinement of the terrain model.

Our earlier work in this overall task area was presented in two previously published papers, one describing basic design issues for this type of system [Hanson, Pentland, & Quam87], and the other providing an overview of our original plans for the implementation [Hanson & Quam88].

4 Automated Detection and Delineation of Cultural Objects in Aerial Imagery

The detection, delineation, and recognition of any significantly broad class of objects (e.g., buildings, airports, cultivated land) in aerial imagery has proven to be an extremely difficult problem. In fact, a nominal component in the solution of this problem, image partitioning, is considered to be one of the most refractory problems in machine vision.

We have recently formulated an optimization-based approach, applicable both to image partitioning and to subsequent steps in the scene analysis process, that involves finding the "best" description of the image in terms of some specified descriptive language.

In the case of image partitioning [Leclerc88, Leclerc89a, Leclerc89b, Leclerc89c, Leclerc89d], we employ a language that describes the image in terms of regions having a low-order polynomial intensity variation plus white noise; region boundaries are described by a differential chain code. The best description is defined as the simplest one (in the sense of least encoding length) that is also stable (i.e., minor perturbations in the viewing conditions should not alter the description). This best description is found using a spatially local and parallel optimization algorithm, that has been implemented on the Connection Machine.

The second step after image segmentation is to simplify the resulting chain-code and polynomial description even further by: (1) describing the boundaries using straight lines and other more global models [Leclerc89c], and (2) grouping nonadjacent regions whose intensity variation can be more simply described by a single polynomial [Leclerc90a, Leclerc90b(*these proceedings*)].

In situations where the required image description must proceed beyond that of a delineation of coherent regions, we require an extended vocabulary relevant to the semantics of the given task. Fua and Leclerc deal with the problem of boundary/shape detection given a rough estimate of where the boundary is located and a set of photometric (intensity-gradient) and geometric (shape-constraint) models for a given class of objects [Fua & Leclerc88, Fua & Leclerc90]. They define an energy (objective) function that assumes a minimal value when the models are exactly satisfied. An initial estimate of the shape and location of the curve is used as the starting point for finding a local minimum of

the energy function by embedding this curve in a viscous medium and solving the dynamic equations. This energy-minimization technique, which evolved from a less-efficient gradient-descent approach [Leclerc&Fua87], has been implemented on the Connection Machine. It has been applied to straight-line boundary models and to more complex models that include constraints on smoothness, parallelism, and rectilinearity, and has been incorporated into the SRI Cartographic Modeling Environment described earlier. In an interactive mode, the user supplies an initial estimate of the boundary of some object (which may be quite complex, like the outline of an aeroplane) and then, if need be, corrects the optimized curve by applying forces to the curve or by changing one of a few optimization/model parameters.

Automatic recognition and delineation of important cartographic objects, such as man-made structures, from aerial imagery has been addressed [Fua&Hanson89a, Fua&Hanson89b]. The basis for the approach is a theoretical formulation of object delineation as an optimization problem; practical objective measures are introduced that discriminate among a multitude of object candidates using a model language and the minimal-encoding principle. This approach is then applied in two distinct ways to the extraction of buildings from aerial imagery: the first is an operator-guided procedure that uses a massively parallel Connection Machine implementation of the objective measure [Fua89] to discover a building in real time given only a crude sketch. The second is an automated hypothesis generator that employs the objective measure during various steps in the hypothesis-generation procedure, as well as in the final stages of candidate selection; both serial and parallel (Connection Machine) approaches are implemented.

We believe that both the Leclerc and the Hanson and Fua techniques represent significant advances in the state-of-the-art in their respective areas of image partitioning and delineation of cultural features. Both systems have been able to produce excellent results in complex situations where existing (typically local) approaches fail. Future work on these techniques will emphasize the incorporation of more complex models, three-dimensional contextual information, and efficient parallel implementations.

5 Object Recognition in the Natural Outdoor World

The natural outdoor environment poses significant obstacles to the design and successful integration of the interpretation, planning, navigational, and control functions of a general-purpose vision system. Many of these functions cannot yet be performed at a level of competence and reliability necessary to satisfy the needs of an autonomous robotic device. Part of the problem lies in

the inability of available techniques, especially those involved in sensory interpretation, to use contextual information and stored knowledge in recognizing objects and environmental features. One of our goals in this effort is to design a core knowledge structure (CKS) that can support a new generation of knowledge-based generic vision systems. A second goal is to actually construct a vision system, which employs the CKS, and has the competence to recognize objects appearing ground level imagery of natural outdoor scenes.

5.1 Core Knowledge System

The CKS is an object-oriented knowledge/database that was originally designed to serve as the central information manager for a perceptual system [Smith&Strat87, Strat&Smith88]. The following facilities of the CKS are of particular importance in supporting the object recognition task.

Multiple Resolution in Space and Knowledge. The CKS employs a multiresolution octree to locate objects only as precisely as warranted by the data. Similarly, a collection of geometric modeling primitives are available to represent objects at an appropriate level of detail. In parallel with the octree for spatial resolution is a semantic network that represents things at multiple levels of semantic resolution.

Inheritance and Inference. The CKS uses the semantic network to perform some limited types of inference that ease the burden of querying the data store. Thus, query responses are assembled not only from those objects that syntactically match the query, but also from objects that can be inferred to match given the relations encoded in the semantic network. Spatial inference is provided based on geometric constraints computed by the octree manipulation routines.

Conflicting Data. One of the realities of analyzing imagery of the real world is that conflicts will result from mistakes in interpretation and from unnoticed changes in the world. The CKS treats all incoming data as the opinions of the data sources, so logical inconsistencies will not corrupt the database. Similarly, values derived through multiple inheritance paths are treated as multiple opinions. This strategy has several advantages and disadvantages. Rather than fusing information as it arises, the CKS has the option of postponing combination until its results are needed. This means that the fusion can be performed on the basis of additional information that may become available, and in a manner that depends on the immediate task at hand. Some information may never be needed, in which case the CKS may forego its combination entirely. The disadvantages are the need to store a larger quantity of data and a slowed response at retrieval time. For an object recognition system like Condor (described below), the CKS seems to provide the right tradeoff.

5.2 Condor: A Contextual Vision System Built on the CKS

Much of the progress that has been made to date in machine vision has been based, almost exclusively, on shape comparison and classification employing locally measurable attributes of the imaged objects (e.g., color and texture). Natural objects viewed under realistic conditions do not have uniform shapes that can be matched against stored prototypes, and their local surface properties are too variable to be unique determiners of identity. The standard machine vision recognition paradigms fail to provide a means for reliably recognizing any of the object classes common to the natural outdoor world (e.g., trees, bushes, rocks, and rivers). In this effort [Strat&Fischler90, *these proceedings*], we have devised a new paradigm which explicitly invokes context and stored knowledge to control the complexity of the decision-making processes involved in correctly identifying natural objects and describing natural scenes.

The conceptual architecture of the system we describe, called Condor (for context-driven object recognition), is much like that of a production system; there are many computational processes interacting through a shared data structure. Interpretation of an image involves the following four process types.

- Candidate generation (hypothesis generation)
- Candidate comparison (hypothesis evaluation)
- Clique formation (grouping mutually consistent hypotheses)
- Clique selection (selection of a "best" description)

Each process acts like a daemon, watching over the knowledge base and invoking itself when its contextual requirements are satisfied. The input to the system is an image or set of images that may include intensity, range, color, or other data modalities. The primary output of the system is a labeled 3D model of the scene. The labels included in the output description denote object classes that the system has been tasked to recognize, plus others from the recognition vocabulary that happen to be found useful during the recognition process. An object class is a category of scene features such as sky, ground, geometric-horizon, etc.

A central component of the architecture is a special-purpose knowledge/database used for storing and providing access to knowledge about the visual world, as well as tentative conclusions derived during operation of the system. In Condor, these capabilities are provided by the Core Knowledge Structure (CKS) as previously discussed.

Visual interpretation knowledge is encoded in *context sets*, which serve as the uniform knowledge representation scheme used throughout the system. The invocation of all processing operations in Condor is governed

by context through the use of various types of context sets: an action is initiated only when one or more of its controlling context sets is satisfied. Thus, the actual sequence of computations, and the labeling decisions that are made, are dictated by contextual information (stored in the Core Knowledge Structure), by the computational state of the system, and by the image data available for interpretation.

The customary approach to recognition in machine vision is to design an analysis technique that is competent in as many contexts as possible. In contrast to this tendency toward large, monolithic procedures, the strategy embodied in Condor is to make use of a large number of relatively simple procedures. Each procedure is competent only in some restricted context, but collectively, these procedures offer the potential to recognize a feature in a wide range of contexts. The key to making this strategy work is to use contextual information to predict which procedures are likely to yield desirable results, and which are not.

Condor operates as follows: For each label in the active recognition vocabulary, all candidate generation context sets are evaluated. The operators associated with those that are satisfied are executed, producing candidates for each class. Candidate comparison context sets that are satisfied are then used to evaluate each candidate for a given class, and if all such evaluators prefer one candidate over another, a preference ordering is established between them. These preference relations are assembled to form partial orders over the candidates, one partial order for each class. Next, a search for mutually coherent sets of candidates is conducted by incrementally building cliques of consistent candidates, beginning with empty cliques. A candidate is nominated for inclusion into a clique by choosing one of the candidates at the top of one of the partial orders. Consistency determination context sets that are satisfied are used to test the consistency of a nominer with candidates already in the clique. A consistent nominee is added to the clique; an inconsistent one is removed from further consideration with that clique. Further candidates are added to the cliques until none remain. Additional cliques are generated in a similar fashion as computational resources permit. Ultimately, one clique is selected as the best semantic labeling of the image on the basis of the portion of the image that is explained and the reliability of the operators that contributed to the clique.

We have taken over 100 photographs (at an experimental site in the foothills behind Stanford University) of which 30 have so far been digitized and successfully processed by Condor. In the future, additional imagery will be acquired and processed to more fully evaluate our approach. Based on our initial experiments, and the unique architecture of our system, we are highly optimistic about the ability of Condor to overcome many of the limitations (with respect to object recognition)

inherent in the traditional machine vision paradigms.

6 Object Modeling from Multiple Images

Our goal in this research effort is to develop automated methods for producing a labeled three-dimensional scene model from image sequences. We view the image-sequence approach as an important way to avoid many of the problems that hamper conventional stereo techniques because it provides the machine with both "redundant" information and new information about the scene. The redundant information can be used to increase the precision of the data and filter out artifacts, the new information can be used for such things as filling in model information along occlusion boundaries and disambiguating matches in the midst of periodic structures.

We have developed two techniques for building three-dimensional descriptions from multiple images. One is a range-based technique that builds scene models from a sequence of range images; the second is a motion analysis technique that analyzes long sequences of intensity images. Our approach for analyzing sequences of range images is to provide the system with a wide variety of object and terrain representations and an ability to judge the appropriateness of these representations for particular sets of data. The variety of representations is required for two reasons. First, it is needed to cover the range of object types typically found in outdoor environments. And second, it is needed to cover the range of data resolutions obtained by a robot vehicle exploring the environment.

In this approach to object modeling an object's description typically goes through a sequence of representations as new data are gathered and processed. One of these sequences might start with a crude blob description of an initially detected object, include a detailed structural model derived from a set of high-resolution images, and end with a semantic label based on the object's description and the sensor system's task. This evolution in representations is guided by a structure we refer to as "representation space", a lattice of representations that is traversed as new information about an object becomes available. One of these representations is associated with an object only after it has been judged to be valid. We evaluate the validity of an object's description in terms of its temporal stability. We define stability in a statistical sense augmented with a set of explanations offering reasons for missing an object or having parameters change. These explanations can invoke many types of knowledge, including the physics of the sensor, the performance of the segmentation procedure, and the reliability of the matching technique. To illustrate the power of these ideas we have implemented a system, which we

call TraX, that constructs and refines models of outdoor objects detected in sequences of range data gathered by an autonomous land vehicle driving cross-country [Bobbitt&Bolles89].

We are continuing to explore the idea of using stability to evaluate the reliability of representations. In addition, we plan to develop new explanations based on support and gravity and to explore ways to combine other types of reliability criteria with stability.

6.1 Building 3-D Descriptions from Image Sequences

We have developed a motion analysis technique, which we call Epipolar-Plane Image (EPI) Analysis [Bolles87]. It is based on considering a dense sequence of images as forming a solid block of data. Slices through this solid at appropriately chosen angles intermix time and spatial data in such a way as to simplify the partitioning problem. These slices have more explicit structure than the conventional images from which they were obtained. In the referenced paper we demonstrated the feasibility of this novel technique for building structured, three-dimensional descriptions of the world.

In later work we extended this technique to locate surfaces in the spatiotemporal solid of data, instead of analyzing slices, in order to maintain the spatial continuity of edges from one slice to the next [Baker&Bolles88]. This surface-building process is the three-dimensional analogue of two-dimensional contour analysis. We have applied it to a wide range of data types and tasks, including medical images such as computed axial tomography (CAT) and magnetic resonance imaging (MRI) data, visualization of higher dimensional (i.e., greater than three-dimensional) functions, modeling of objects over scale, and assessment in fracture mechanics.

We have also implemented a version of EPI analysis that works incrementally, applying a Kalman filter to update the three-dimensional description of the world each time a new image is received [Baker&Bolles88]. As a result of these changes the program produces extended three-dimensional contours instead of sets of isolated points. These contours evolve over time. When a contour is initially detected, its location is only coarsely estimated. However, as it is tracked through several images, its shape typically changes into a smooth three-dimensional curve that accurately describes the corresponding feature in the world.

Recently we have extended of the EPI analysis technique in two directions. The first is the modeling of biological structures from tomographic data [Baker90]. The descriptive formalism we are developing models tissue as two-dimensional manifolds in three space. We have used this type of model to demonstrate simple versions of surgical simulation, kinematic modeling, and kinematic analysis. In the second extension we are using

the temporal tracking mechanism in EPI analysis to detect and track moving objects from moving sensors. We have added evaluation routines that select key features to be tracked on the moving objects. We are currently exploring techniques for constructing three-dimensional descriptions of the tracked objects.

6.2 Detecting Moving Objects from Moving Sensors

Building upon our work in motion vision and terrain modeling, we have recently begun development of techniques for detecting and tracking moving objects from a moving platform. This work is being performed jointly with the Machine Vision Group at the David Sarnoff Research Center.

Motion (in a sequence of images) provides one of the strongest cues available about the presence of a possible target in a scene. However, when a sensor is moving, everything in the image is moving. Therefore, detection of possible targets requires separating the motion induced by the movement of the sensor from the motion caused by the movement of the target. One approach to this problem, which has been developed at Sarnoff and other places, is to model the "background" image flow as a simple parametric flow field, and then use this model to eliminate image motion consistent with that flow. Any motion not consistent with the background movement is labelled as a possible moving object. Of course, such an approach fails dramatically when the simple background assumption is violated (e.g., when the terrain contains many ridges and valleys, which induce a wide variety of background image motion).

The approach we are taking to handle these complex backgrounds is to integrate a full three-dimensional terrain map into the target detection system. The basic idea is to (i) use the model of the terrain and the known motion of the sensor to predict the motion observed by the sensor, (ii) compute the actual motion present in the imagery, and (iii) use the differences to robustly detect and track moving targets. We expect that the addition of a terrain model will yield a significantly more robust and sensitive detection and tracking system than those relying on simpler background assumptions.

Note, however, that inaccuracies in the terrain model could produce differences between the predicted and computed image motion that, over a small number of images, look similar to moving objects. Therefore, integral to this approach is the ability to correct an a priori model of the environment as new data are acquired. We plan to draw upon current techniques for recovering structure from motion to dynamically update our models. By continually improving the underlying model, the motion detection procedure will be able to distinguish short-term deviations from moving objects.

As part of our research strategy we have decided to

test our algorithms on both simulated and real data. (We have used the Cartographic Modeling Environment to provide extensive simulation data.) The advantage of simulated data is that we know "ground truth" and therefore are in a better position to judge the competence of the algorithms (along some key dimensions) than when we analyze real data. This strategy has already paid off. Our initial experimentation with simulated data pointed out a serious weakness in displaying warped images to demonstrate the results of optic flow computations. At occlusion boundaries optic flow techniques locate matches (and compute flow vectors) for points that have similar greyscale values. This procedure leads to stabilized intensity images when the flow vectors are used to warp one image into another, but the flow vectors are incorrect. Given a terrain model, we are now able to predict occlusion boundaries and avoid these erroneous results.

7 Acknowledgment

The following researchers have contributed to the work described in this report: H.H. Baker, S.T. Barnard, A. Bobick, R.C. Bolles, O. Firschein, M.A. Fischler, P. Fua, M.J. Hannah, A.J. Hanson, Y. Leclerc, L.H. Quam, G.B. Smith, T.M. Strat, R.S. Szeliski, and H.C. Wolf.

8 References

The following recent publications result fully or in part from SRI's image-understanding research program.

1. Baker, H.H., "Building Surfaces of Evolution: The Weaving Wall," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.
2. Baker, H.H., "Surface Reconstruction from Image Sequences," invited presentation, *11th European Conference on Visual Perception*, Bristol, U.K., August 1988.
3. Baker, H.H., "Generalizing Epipolar-Plane Image Analysis on the Spatiotemporal Surface," *IEEE Conference on Computer Vision and Pattern Recognition*, Ann Arbor, Michigan, June 1988.
4. Baker, H.H. and R.C. Bolles, "Generalizing Epipolar-Plane Image Analysis on the Spatiotemporal Surface," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.
5. Baker, H.H., R.C. Bolles, and D.H. Marimont, "Generalizing Epipolar-Plane Image Analysis for Non-Orthogonal and Varying View Directions," *Proc. DARPA Image Understanding Workshop*,

- University of Southern California, Vol.II, pp.843-848, February 1987.
6. Baker, H.H., "Reimplementation of the Stanford Stereo System: Integration Experiments with the SRI Baseline Stereo System," Tech. Note 431, Artificial Intelligence Center, SRI International, Menlo Park, California, February 1989.
 7. Baker, H.H., "Surface Modeling with Medical Imagery," *Proc. NATO Advance Workshop on 3D Imaging in Medicine*, Travemunde, West Germany, June 1990.
 8. Barnard, S.T., "Stochastic Stereo Matching Over Scale," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.
 9. Barnard, S.T., "Stereo Matching by Hierarchical, Microcanonical Annealing," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.II, pp. 792-796, February 1987.
 10. Barnard, S.T., "Stochastic Stereo Matching on the Connection Machine," in *Proc. of Fourth International Conference on Supercomputing*, Santa Clara, California, 30 April - 5 May 1989.
 11. Barnard, S.T., "Stochastic Stereo Matching on the Connection Machine," *Proc. of Image Understanding Workshop*, Palo Alto, California, pp. 1021-1031, May 1989.
 12. Barnard, S.T., "Stochastic Stereo Matching Over Scale," *Int. J. of Computer Vision*, 3(1)1989.
 13. Barnard, S.T., "Recent Progress in CYCLOPS: A System for Stereo Cartography," *these proceedings*.
 14. Barnard, S.T. and M.A. Fischler, "Computational and Biological Models of Stereo Vision," to appear in Wiley Encyclopedia of AI, 2nd ed; also, *these proceedings*.
 15. Bobick, A.F. and R.C. Bolles, "Representation Space: An Approach to the Integration of Visual Information," *Proc. of Image Understanding Workshop*, Palo Alto, California, pp. 263-272, May 1989.
 16. Bobick, A.F. and R.C. Bolles, "Representation Space: An Approach to the Integration of Visual Information," *Proc. of CVPR*, San Diego, California, pp. 492-499, June 1989.
 17. Bolles, R.C., H.H. Baker, and D.H. Marimont, "Epipolar-Plane Image Analysis: An Approach to Determining Structure from Motion," *International Journal of Computer Vision*, Kluwer Academic Publishers, Vol.I, No.1, pp. 7-5, June 1987.
 18. Bolles, R.C. and A.F. Bobick, "Exploiting Temporal Coherence in Scene Analysis for Autonomous Navigation," *Proc. Robotics and Automation Conference*, Scottsdale, Arizona, May 1989.
 19. Bobick, A.F. and R.C. Bolles, "An Evolutionary Approach to Constructing Object Descriptions," *Proc. of 5th International Symposium of Robotics Research*, Tokyo, Japan, pp. 172-180, August 1989.
 20. Corby, N.R., Mundy, J.L., Vrobel, P.A., Hanson, A.J., Quam, L.H., Smith, G.B., and T.M. Strat, "PACE - An Environment for Intelligence Analysis," *Proc. DARPA Image Understanding Workshop*, Boston, Massachusetts, pp. 342-350 (April 6-8, 1988).
 21. Firschein, O. and M.A. Fischler, "AI Application of Supercomputers: The Vision Problem," *Fourth International Conference on Supercomputing*, Santa Clara, California, April 30-May 5, 1989.
 22. Fischler, M.A. and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. ACM*, Vol.24, No. 6, pp. 381-395, June 1981.
 23. Fischler, M.A. and O. Firschein, *Intelligence: the Eye, the Brain, and the Computer*, Addison-Wesley, Reading, Massachusetts, 1987.
 24. Fischler, M.A. and O. Firschein, *Readings in Computer Vision*, Morgan-Kaufmann, Los Altos, California, 1987.
 25. Fischler, M.A. and O. Firschein, "Parallel Guessing: A Strategy for High-Speed Computation," *Pattern Recognition*, Vol.20, No.2, pp. 257-263, 1987.
 26. Fischler, M.A. and T.M. Strat, "Recognizing Objects in a Natural Environment: A Contextual Vision System (CVS)," *Proc. of Image Understanding Workshop*, Palo Alto, California, pp. 774-796, May 1989.
 27. Fischler, M.A., "An Overview of Computer Vision Research at SRI International - Themes and Progress," *International Journal of Computer Vision*, Vol. 3, No. 1, 1989.
 28. Fua, P. and A.J. Hanson, "Using Generic Geometric Models for Intelligent Shape Extraction," *Proc. DARA Image Understanding Workshop*, University of Southern California, Vol.I, pp. 227-233, February 1987.
 29. Fua, P. and Y. Leclerc, "Finding Object Boundaries Using Guided Gradient Ascent," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.II, pp. 888-891, February 1987.

30. Fua, P.V. and A.J. Hanson, "Extracting Generic Shapes Using Model-Driven Optimization," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.
31. Fua, P.V. and Y.G. Leclerc, "Model Driven Edge Detection," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.
32. P. Fua and A.J. Hanson, "Objective Functions for Feature Discrimination: Theory," *Proc. DARPA Image Understanding Workshop*, April 1989.
33. P. Fua and A.J. Hanson, "Objective Functions for Feature Discrimination: Applications to Semi-automated and Automated Feature Extraction," *Proc. DARPA Image Understanding Workshop*, April 1989.
34. P. Fua, "Object Delineation as an Optimization Problem, A Connection Machine Implementation," in *Proc. Fourth International Conference on Supercomputing*, Santa Clara, California, 30 April - 5 May 1989.
35. P. Fua and Y.G. Leclerc, "Model Driven Edge Detection," *Machine Vision and Applications*, in press.
36. Hannah, M.J., "SRI's Baseline Stereo System," *Proc. DARPA Image Understanding Workshop*, Miami Beach, Florida, pp. 149-155, December 9-10, 1985.
37. Hannah, M.J., "Test Results from SRI's Stereo System," *Proc. DARPA Image Understanding Workshop*, Cambridge, Massachusetts, pp. 740-744, April 1988.
38. Hannah, M.J., "Digital Stereo Image Matching Techniques," *International Archives of Photogrammetry and Remote Sensing*, Vol. 27-III, pp. 280-293, XVth ISPRS Congress, Kyoto, Japan, July 1988.
39. Hannah, M.J., "A System for Digital Stereo Image Matching," *Photogrammetric Engineering and Remote Sensing*, Vol. 55, No. 12, December 1989, pp. 1765-1770.
40. Hanson, A.J., A.P. Pentland, and L.H. Quam, "Design of a Prototype Interactive Cartographic Display and Analysis Environment," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.II, pp. 475-482, February 1987.
41. Hanson, A.J. and L. Quam, "Overview of the SRI Cartographic Modeling Environment," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.
42. Hanson, A.J., "Hyperquadrics: Smoothly Deformable Shapes with Convex Polyhedral Bounds," *Computer Vision, Graphics and Image Processing* 44, 191-210, 1988.
43. Hanson, A.J. and L. Quam, "A Cartographic Visualization Environment," in *Proc. of the Military Computing Conference*, Anaheim, California, pp. 233-240, May 3-5, 1988.
44. Heeger, D. J., "Optical Flow from Spatiotemporal Filters," *Proc. First International Conference on Computer Vision*, London, England, pp. 181-190, June 8-11, 1987.
45. Leclerc, Y.G., and P.V. Fua, "Finding Object Boundaries Using Guided Gradient Ascent," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.II, pp. 888-891, February 1987.
46. Leclerc, Y.G., "Constructing Simple Stable Descriptions for Image Partitioning," *Proc. DARPA Image Understanding Workshop*, Cambridge, Massachusetts, pp. 365-382, April, 1988.
47. Leclerc, Y.G., "Constructing Simple Stable Descriptions for Image Partitioning," *International Journal of Computer Vision*, Vol. 3, No. 1, 1989a.
48. Leclerc, Y.G., "Segmentation via Minimal-Length Encoding on the Connection Machine," *Fourth International Conference on Supercomputing*, Santa Clara, California, April 30-May 5, 1989b.
49. Leclerc, Y.G., "Image and Boundary Segmentation via Minimal-Length Encoding on the Connection Machine," *Proc. DARPA Image Understanding Workshop*, Palo Alto, California, pp. 1056-1069, May, 1989c.
50. Leclerc, Y.G., "The Local Structure of Image Intensity Discontinuities," Ph. D. Thesis, McGill University, Montréal, Québec, Canada, May, 1989d.
51. Leclerc, Y.G., "Simplicity of Description as the Basis for Visual Interpretation," Working Notes of the AAAI Spring Symposium on The Theory and Application of Minimal-Length Encoding, Stanford University, Palo Alto, California, March, 1990a.
52. Leclerc, Y.G., "Region Grouping using the Minimum-Description-Length Principle," *Proc. DARPA Image Understanding Workshop*, 1990b, *these proceedings*.
53. Smith, G.B. and T.M. Strat, "Information Management in a Sensor-Based Autonomous System," *Proc. Image Understanding Workshop*, University of Southern California, Vol.I, pp. 170-177, February 1987.

54. Strat, T.M. and M.A. Fischler, "A Context-Based Recognition System For Natural Scenes and Complex Domains," Proc. DARPA Image Understanding Workshop, 1990, *these proceedings*.
55. Strat, T.M. and G.B. Smith, "Core Knowledge System: Storage and Retrieval of Inconsistent Information," Proc. DARPA Image Understanding Workshop, Washington, D.C., April 1988.

Image Understanding at the GRASP Laboratory

Ruzena Bajcsy*

GRASP Laboratory

Computer and Information Science Department

University of Pennsylvania

Philadelphia, PA 19104

Abstract

Research in the GRASP Laboratory has two main themes, parameterized multi-dimensional segmentation and robust decision making under uncertainty. The multi-dimensional approach interweaves segmentation with representation. The data is explained as a best fit in view of parametric primitives. These primitives are based on physical and geometric properties of objects and are limited in number. We use primitives at the volumetric level, the surface level, and the occluding contour level, and combine the results. The robust decision making allows us to combine data from multiple sensors. Sensor measurements have bounds based on the physical limitations of the sensors. We use this information without making a priori assumptions of distributions within the intervals or a priori assumptions of the probability of a given result.

1 Introduction

Our basic approach to Image Understanding can be summarized as follows: we seek to divide the observed scenes into 3D objects. 2D images are observations/measurements of these 3D physical objects under certain illuminations and perspective projections. Hence, the process of Image Understanding includes the transformation of the 2D data into a description of 3D objects in terms of physical and geometric primitives. This grouping of image data, to produce such a description, is called **segmentation**.

Though image segmentation has been treated separately from shape representation in the past, solving the two problems separately is very difficult. If an image is correctly divided into 3D components, ambiguities in the 2D segmentation can be resolved more easily. Conversely, a range image can be partitioned more easily into 3D shapes given a good 2D segmentation of the same scene. Indeed, dividing the scene into 3D primitives is

a form of segmentation. Hence, 2D image segmentation and 3D description of parts should act together as a cooperative process [Bajcsy *et al.*, 1990], a combined 2D-3D segmentation process.

Range images and 2D images complement each other. Range data is a reflection of geometric properties of the objects only. However, 2D images contain indirect information about geometric properties in combination with surface properties (color, texture, translucence, etc.), which is not available at all in the range data. Because of this, a cooperative 2D/3D segmentation process has the potential to be a major improvement over separate segmentation.

By extension, the grouping of data through time is also segmentation. One aspect of grouping temporal data is the construction of range images via sliding stereo or structured light range images. Another aspect of grouping temporal data is the understanding of movable or removable parts in objects in a scene. Understanding that a part moves as a unit, separately from another part, is a partitioning of the data. Without a priori knowledge, this cannot be detected by noncontact sensing. This segmentation would allow understanding of the dynamic properties, the mechanical properties and the kinematic properties of objects. However, in this paper, we concentrate on the aspects of 2D/3D segmentation process which can be accomplished through non-contact sensing.

1. Definition of the segmentation problem:

- Segmentation is **partitioning** the space into meaningful parts. This can be done on intensity images or range images.
- Segmentation is **data reduction** and **requantization** of sensory measurements into some primitive elements.
- Segmentation is an **inference of a symbolic description** of the world from one or more images.

2. What kind of measurements are available in non-contact sensing?

- **Multispectral Image:** camera with filters having different spectral sensitivities. We measure energy flux incident on one image plane (irradiance) that combines the following components:
 - energy and spectral distribution of the incident light,

*Acknowledgement: Navy Grant N0014-88-K-0630, AFOSR Grants 88-0244, AFOSR 88-0296; Army/DAAL 03-89-C-0031PRI; NSF Grants CISE/CDA 88-22719, IRI 89-06770; and Dupont Corporation

- reflectance properties of the objects, and
- geometry - orientation of the surface with respect to the viewer and the light source.
- **Multiple Views:** In the paradigm of Active Vision, we seek the appropriate place to look, including the movement of the observer. Motion of the observer allows us to construct Range Images. In principle, this could be done using stereo; however, in this work we use structured light range imaging.

3. What kind of primitive elements should we use?

- Primitives must balance the trade-offs between:
 - data reduction versus faithfulness to measured data and
 - localness versus globalness.
- Primitives should correspond to segments in terms of **physical phenomena**.

Physical phenomena are manifested in the image via the following discontinuities: depth, orientation, albedo, shadow, shading, and specularity. Hence, our segmentation and resulting descriptions will be in terms of physical properties of the world (surface reflectance, shading, shadow, highlights and geometry) rather than in terms of image attributes.

In Section 2 our work detecting highlights and inter-reflections is described. Section 3 details segmentation of images into objects/surfaces made of specular and diffuse materials [Bajcsy *et al.*, 1989]. Section 4 explains the segmentation of the range image and/or lightness image into surfaces. Section 4 represents our efforts to recover underlying geometric structure from an image. Since geometry involves more than just surfaces [Leonardis *et al.*, 1990], we find volumetric descriptions in terms of superquadric parts. All of this processing is only from one viewing angle, which is not sufficient for describing a scene composed of opaque objects. This question of where to go next, based on the first view, is described briefly in Section 5 and in more detail in the paper by Maver and Bajcsy in this Proceeding.

The above work is being extended into the development of a formal model for an observer of an indoor scene being explored by a mechanical hand [Bajcsy and Sobh, 1990]. The task for the observer is to have a full view of the hand and the object being held and/or manipulated. We have adopted the formalism of discrete event dynamic systems (DEDS), which allows us to predict under which conditions the observer-automaton can accomplish the task. This includes both the stability and the observability conditions. The formalism of DEDS has been described in [Ho, 1987], [Ramadge and Wonham, 1987b; Ramadge and Wonham, 1987a] and others.

Observing and understanding motion is an established part of the Image Understanding effort. In the GRASP Laboratory, we have used a temporally-oriented approach rather than a spatially oriented approach. This approach is described in Section 6, and detailed in the paper by Wohn and Iu in this Proceeding.

The underlying theory of updating procedures and decision making under uncertainty is summarized in sec-

tion 7, and described more thoroughly in the paper by Mintz, McKendall and Kamberova in this Proceeding.

2 Color Image Understanding: Image Segmentation and Detection of Highlights and Inter-Reflections Using Color.

Color image segmentation should be based on changes in object colors. In addition to the object color changes, however, an image of three-dimensional real objects contains variations because of shading, shadows, highlights and inter-reflections. Detection and separation of highlights for successful segmentation has been the focus of recent efforts in color image segmentation [Gershon, 1987] [Klinker *et al.*, 1988]. We have approached the construction of a computational model for color image segmentation not only with detection of highlights, but also with detection of small color changes induced by inter-reflections.

We use the dichromatic model [Shafer, 1985] for dielectric materials. There are two reflection mechanisms—surface reflection and body reflection. The surface or interface reflection occurs at the interface of air and object material, and can be specular and/or diffuse reflections depending on the surface roughness. Surface reflectance at dielectric materials are spectrally flat for both specular and diffuse reflections. Body reflection, on the other hand, is spectrally colored depending on the pigments, and always diffused. When the body reflectance is non-flat, we can detect the flat component of surface (or interface) reflectance regardless of surface roughness. Highlights are caused by specular surface reflection; inter-reflections between the objects are caused by both surface (specular or diffuse) and body reflections.

To better represent and process the image color, a color metric space is developed based on the physical model of the camera and filters. The measured color in R,G,B space is transformed into the metric space using a set of orthogonal basis functions. We used the first three of Fourier basis functions. Within our framework, however, any orthogonal basis functions can be used for better representation of natural colors [Cohen, 1964] [Judd *et al.*, 1964]. The metric space is similar to the opponent space in human vision with intensity, hue and saturation. With orthogonal values, we can manipulate each component of color separately or in combinations. Transformation from R,G,B into orthogonal values is a pixel parallel process, and is implemented on a Connection Machine (CM2a).

Since illumination is usually spectrally colored, calibration of measured images is performed with a white object of reference to whiten the illumination. Whereas the spectral distribution of object surfaces is not changed by shading and shadow, it is affected by highlights and inter-reflections between the objects even under the white illumination. Since the highlights add the whiteness to the object color under the whitened illumination, they can be detected by observing the change of saturation in the uniformly colored objects, which is equiva-

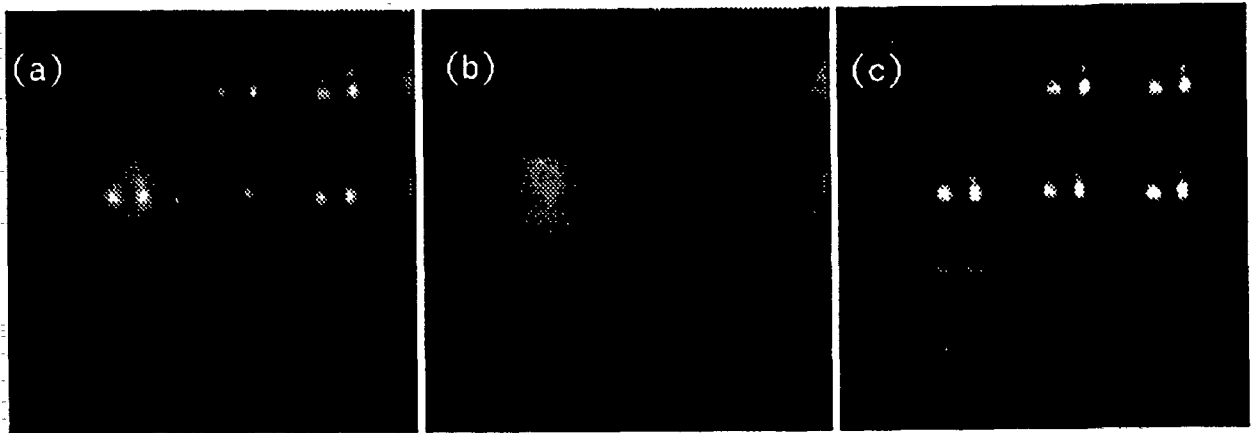


Figure 1: **a** Intensity **b** Body Reflections, and **c** Surface Reflection. In (c), inter-reflections are visible as spatially diffused low intens' values. The bright peaks of highlights are strong specular surface reflections.

lent to examining to detect any spectrally flat reflectance added in the body color. The inter-reflections are also detected with the change in saturation and in hue values.

Segmentation is carried out in two steps; hue and saturation segmentation under the assumption that objects are piecewise uniform in hue and saturation. Intensity is not easy to use in the presence of shading, shadow and highlights. Illumination whitening is important for hue segmentation since under white illumination, the highlights do not shift the hue values of object colors. Highlights are detected by observation of saturation values. The use of the reference plate is not necessary when the illumination is weakly colored. Roughly detected highlights can be used for the reference. Within each region, the detection and separation of highlights and inter-reflections is highly parallel, since it is accomplished via thresholding and arithmetic operations.

Our method is improved over previous works [Gershon, 1987] [Klinker *et al.*, 1988], in a few significant ways. The previous methods can detect strong surface specular reflection, but are not reliable in detecting small surface reflections that are diffused. Since we interpret the reflection mechanisms in our color space with all the spectral characteristics of sensors considered, we can better observe the spectral variation of reflection. Therefore, we can detect not only strong and distinctively appearing specular reflection, but also small surface reflections which are spatially diffused. Inter-reflections usually have a diffused appearance. The inter-reflections can be detected by the change in saturation values although hue values also change.

Figure 1 (b) shows Body Reflections and (c) shows the Surface Reflections of which the original intensity is shown in Figure 1 (a). Though inter-reflections are barely visible in the original color image, they are not visible in Figure 1 (a). These inter-reflections are visible in figure 1c, as spatially diffused, low-intensity values on the horizontal strip.

In our previous work, we have concentrated on spectral information. We are expanding our investigation to include:

1. Multi-dimensional segmentation. The spatial seg-

mentation results can be used wtogether with segmentation by hue and saturation. [Leonardis *et al.*, 1990].

2. Active Camera Movements. This will allow us to observe different moving patterns of surface and body reflections as the observer moves.
3. Changing illumination (when possible) to disambiguate the image variations due to the change of shading/shadow and albedo.

3 Segmentation as the Search for the best Description of the Image in terms of Primitives.

A new paradigm for image segmentation has been developed. We segment images into piecewise continuous patches [Leonardis *et al.*, 1990]. Data aggregation is performed via model recovery in terms of variable-order bivariate polynomials using iterative regression. All the recovered models are potential candidates for the final description of the data. Selection of the models is achieved through a maximization of quadratic Boolean problem. The procedure can be adapted to prefer certain kinds of descriptions (one which describes more data points, or has smaller error, or has lower order model). We have developed a fast optimization procedure for model selection. The major novelty of the approach is in combining model extraction and model selection in a dynamic way. Partial recovery of the models is followed by the optimization (selection) procedure where only the "best" models are allowed to develop further. The results obtained in this way are comparable with the results obtained when using the selection module only after all the models are fully recovered, while the computational complexity is significantly reduced. We test the procedure on real range and intensity images.

We believe this segmentation schema is a tool that will prove useful in many tasks of early vision. The two procedures (model recovery and recover-and-select) clearly show that the whole can be greater than the sum of its parts (synergism). The iterative approach combin-

ing data classification and model fitting shows that segmentation and modeling are not two independent procedures but have to be integrated. The procedure which dynamically combines model recovery with model selection proves to be much more efficient than applying the modules one after another.

Another important conclusion that we have drawn from our work is that reliable segmentation can only be achieved by considering many competitive solutions and choosing those which reveal some kind of structure in terms of underlying models. Fine-tuning of feature detectors does not lead to reliable segmentation, because of the variability of the input data. Initial local estimates, no matter how good they are, do not necessarily lead to a good result, and more global information is needed. Optimization performed on the level of primitives rather than on a pixel level not only improves the performance enormously in terms of computational complexity but also gives more reliable results.

The results are grouped such that the top row of the figure (from left to right) shows the original image, its 3-D perspective plot, the reconstructed image from the piecewise continuous segmented patches, and the 3-D plot of the reconstructed image. The range images are displayed with the depth value at each pixel from a reference plane appearing larger if the pixel is closer to the camera. The white square in the patch indicates the seed region for that patch. The individual surface patches are displayed in the second row of the figures in the order in which they were selected by the model selection procedure, and are referred to below with their position in the row, counting from left to right.

The Coffee-mug image: The convex and concave portions of the body of the cup are recovered as individual second-order patches, as shown in the first two images of the bottom row in figure 2. The handle consists of very curved patches which are modeled piecewise for the given scale (which directly relates to the compatibility constraint). According to the results, the missing parts are better described as individual pixels than as parametric patches (due to the scale consideration). It should be noted that the jump (C_0) discontinuities are clearly delineated by the neighboring regions.

4 Integrated approach to 3D shape (volume, surface, contour) recovery via parametric descriptions

In section 3, we described segmentation of a 2-1/2 D image into surface patches. In other work [Solina and Bajcsy, 1990] we have used parametric descriptions of superquadrics to fit volumetric aspect of a shape. We are now in the process of developing a paradigm for decomposition of complex objects in range images into the constituent parts based on the shape, using contour, surface, and volumetric primitives [Gupta and Bajcsy, 1990]. Unlike previous approaches, we use geometric properties derived from both boundary-based (surface contours and occluding contours), and primitive-based (biquadric patches and superquadric models) representations to de-

fine and recover part-whole relationships, without a *priori* knowledge about the objects or the object domain. The descriptions thus obtained are independent of position, orientation, scale, domain and domain properties, and are based purely on geometric considerations. Since both boundary-based and primitive-based primitives are included in our vocabulary, the representation is expressive and robust.

In the computer vision literature, the partitioning of images and description of individual parts is called segmentation and shape representation respectively. We have presented arguments in Bajcsy, Solina, and Gupta [Bajcsy *et al.*, 1990] that the problems of segmentation and representation are related and *must be* treated simultaneously. We propose that for obtaining a global shape description from single-viewpoint 3-D data requires addressing shape at the following levels :

1. **Volumetric level:** Superquadric shape primitives capable of modeling parts in three dimensions are needed to describe global 3-D shape of parts.
2. **Surface level:** Surface primitives describe *internal surface boundaries* and *surface patches* which are difficult to model by volumetric primitives, but present a more accurate and detailed description of shape that is neither too local nor too global.
3. **Occluding Contour level:** The Occluding contour encodes the 3-D shape of parts projected on the image plane.

Given the three different modules for extracting volume, surface and boundary properties, how should they be invoked, evaluated and integrated? To incorporate the best of the coarse to fine and fine to coarse segmentation strategy, we perform volume, surface, and boundary fitting in parallel on the input data. This requires evaluation and comparison of information embedded in models built by different aggregation methods. The occluding contour is segmented into parts at concavities and convexities using the classical techniques. The surface is segmented into planar and bi-quadric patches using the segmentation algorithm outlined in the previous section. The segmentation also gives reliable internal C_1 (orientation) discontinuities, which are vital for part-segmentation, but are very difficult to localize using standard edge-detection techniques. The superquadric model, being an object centered global part-model, is not amenable to such segmentation techniques. So the problem of fitting or recovering the superquadric models to parts of an object has to be attempted in such a way as to make use of the segmentation information from the surface and contour models that provide local segmentation at their respective levels. The superquadric model recovery itself proceeds from coarse to fine (global to local), generating residuals and hypotheses about volumetric parts as described below. We are developing a control module to accomplish this non-trivial task in a systematic manner.

To satisfy the practical constraints of computability and robustness, we pose the problem of integration in terms of evaluation of the intermediate descriptions and segmentation of the objects in a closed loop process. To

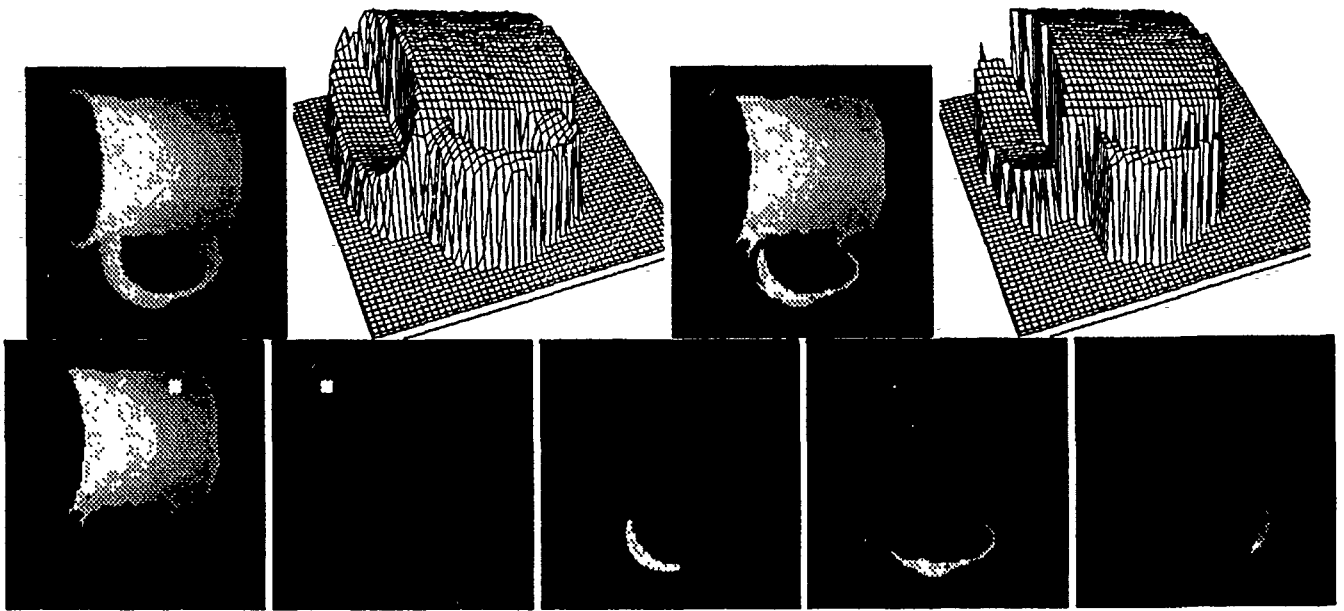


Figure 2: The coffee-mug Image: The highly curved handle is modeled as a combination of the smaller patches.

evaluate the superquadric models, we have developed a set of quantitative and qualitative measures, that generate global and local residuals of the models [Gupta *et al.*, 1989]. The qualitative residuals are the regions of *underestimation*, and surface and contour *overestimation*. The fundamental principle behind our approach is that these residuals are generated because of the presence of parts (or negative volume), otherwise a correct volumetric model would be obtained during the initial global fit. The regions of contour overestimation guide the localization of the concavities in the occluding contour. The concavities in the occluding contour are in turn used to constrain the superquadric model to fit only a part and not the complete object [Gupta and Bajcsy, 1990]. It essentially provides topological constraints to restrict the model to a part of the object. Although the surface segmentation is complete at the surface level, it does not impose any topological constraints needed for superquadric level of part segmentation. In addition, a mechanism to combine individual surface patches to form the volumetric parts is needed to generate strong hypotheses about potential superquadric models. We have observed that the information from the regions of underestimation (primarily caused due to the parts 'sticking out') makes natural clusters of surface patches forming volumetric parts. Using this simple observation, we have obtained encouraging results on various complex objects.

Our goal is to develop a general-purpose shape segmentation paradigm to generate object-centered and view-invariant descriptions from a single general view of complex objects. The approach has applications in object localization and recognition, automatic model generation, and domain specific high level tasks.

5 Searching for Additional Information

The task of constructing a volumetric description of a scene from a single image is an underdetermined problem, whether it is a range image or an intensity image. Once the first image has been taken, we develop a strategy to get the additional information that will allow us to complete the volumetric description.

Range images are $2\frac{1}{2}$ D images, where the value of a pixel corresponds to the distance from the observer to the closest point in the scene. Some parts of the scene may be partially occluded by objects. In structured light range images, a pixel with no data corresponds to an occluded area.

At the GRASP Laboratory, we have two structured light range imaging systems. One uses a fixed laser (providing the plane of light), while the objects move on a linear stage. Though this system provides high quality range data, it is limited to a single view of the scene. However, our second system uses a laser-camera pair, mounted on a Puma 560 robot arm, which moves over the scene [Tsikos, 1989]. A range image is acquired by moving the laser-camera pair along a straight line in space, aiming at the scene. It is capable of viewing a scene within the robot workspace from many different angles. This capability allow us to construct a complete 3D model of a scene.

Our strategy is to use the information in a narrow zone around the occluded regions. Occluded regions are approximated by polygons. From the height of the border of the occluded regions and from the geometry of the edges of the polygonal approximation, we calculate the direction which will most effectively show us what is in the occluded area. Experimental results on range data are described in the paper by Maver and Bajcsy in this

6 Image Motion Analysis: A Temporally-oriented Approach

The fact that the relative motion between the viewer and the object can be recovered from a sequence of images is well-known, and articles on the subject are abundant. The majority of previous work dealt with the existence and the uniqueness of solution when the input was assumed to be given in the proper form (image flow field, disparity vectors, position of feature points, conic contours, etc). A typical result states that there are K solutions for the 3-D motion and the object structure, given M features over N frames. For the uniqueness, some suggest that K could be 1 if M' (where $M' > M$) features are used; others suggest that K could be 1 if N' (where $N' > N$) frames are used, etc. While these results provide us with some useful theoretical framework, all known algorithms derived from their constructive proofs have turned out to be very sensitive to the input noise [Tsai and Huang, 1984; Waxman and Ullman, 1985].

In estimating 2-D motion, there is no known algorithm that estimates the 2-D image motion with sufficient accuracy for the 3-D motion recovery algorithm. For example, the image flow estimated by any of the well-known techniques [Hildreth, 1984], [Horn and Schunck, 1981] does not seem to be useful at all for the purpose of 3-D motion recovery. Furthermore, the accuracy of image motion estimate is lower-bounded: Even for a noise-free, synthetically generated image sequence, there is the discretization effect in spatial as well as temporal domains. Sometimes this discretization effect alone exceeds the maximal noise level the 3-D motion recovery algorithm can tolerate.

The error in 2-D motion amplifies the error in 3-D motion parameters. The exact nature of error propagation depends not only on the specific algorithm but also on many factors such as the viewing angle of camera, and motion and structure parameters themselves, and thus it is hard to derive the error formula analytically. Unless one imposes unrealistic assumptions such as a huge viewing angle (typical viewing angle of camera lens is 30-50 degrees and the object of interest occupies even a smaller field of view), all the existing algorithms perform poorly under the presence of realistic noise. The difficulty is that the 3-D motion recovery is ill-conditioned.

Hence, a more realistic approach, as far as the recovery of 3-D motion is concerned, is to improve the motion parameters over time, under the presence of noise. Just like any other physical system, the entire process from the image sequence to the 3-D motion may be viewed as a dynamic system, and the problem can be formulated as a non-linear state estimation problem.

Our approach is not merely meant to attempt to improve the motion estimates over time. Most of the previous algorithms mentioned above are "biased" more toward the spatial information in image sequences than the temporal information, in the following sense. In the token matching approach, the main idea is to find the

minimal number of points that guarantee a unique solution when a small number (typically 2 or 3) of frames are given. Similarly, the optical flow approach seeks the lowest order of flow derivatives from a single snapshot of optical flow field. Of course, both "instantaneous" approaches have to make use of the temporal information in one way or another, but the extent of temporal information being used is fixed at the stage of problem formulation. We may call these approaches *spatially-oriented*.

Our approach first explores the temporal information prior to the usage of the spatial one. Here, a typical question one may ask is: "How many frames are needed when N features are given?", as opposed to "How many features are needed when M frames are given?". We call it *temporally-oriented approach* (TOA). The importance of TOA's is that we can avoid many problems which one may encounter in the SOA's. Since we observe motion over an extended time interval, we can reduce the number of features that are used in the computation. In fact, we have shown that we could even recover the 3-D motion of a single particle. Consequently, the problem of requiring multiple features can be eliminated and the task of segmentation is thereby reduced. Further, as we use more frames to estimate the 3-D motion, the problem itself becomes more well-conditioned. When we observe a moving object such as a space shuttle or a baseball, the longer we observe, the more accurately we can estimate its motion and predict its position. In TOA's, we rely on the temporal information from the moving object while keeping the amount of spatial data in a single image as small as possible. Of course we may use multiple features to get a more robust estimate if they are available. Therefore, the TOA's and the previous multi-frame approach are different in their motivation.

7 Robust Multi-sensor Fusion

We have developed a coherent methodology for fusing data from multiple sensors in uncertain environments. Since sensors exhibit noisy behavior that cannot be eliminated completely, all sensor measurements are uncertain. However, sensor errors can be modeled statistically and geometrically, using both physical theory and empirical data. For example, multiple range sensors may be located on the wings of an aircraft. As the wings themselves flex, the precise location of the sensors move, relative to the center of the aircraft. These sensor location errors can be bounded, but it is not necessarily desirable to characterize these errors statistically. Also, sensors may break or become worn out, but still send data. For example, a camera with a broken IR filter still sends data, which seems to be within the dynamic range of the system (though perhaps at the saturation limit). This kind of uncertainty can be handled by a statistical decision theoretic approach.

A single distribution is usually an inadequate description of sensor noise behavior. It is much more realistic and much safer to identify an envelope or class of distributions, one of whose members could represent the actual statistical behavior of the given sensor. For example, it may be difficult to fit the error distribution of a

malfunctioning sensor by a Gaussian distribution; however, ϵ -contamination models provide a good alternative. Other reasons for uncertainty in statistical sensor models include: sporadic interference, drift due to aging, temperature variations, miscalibration, quantization, and other significant nonlinearities over the dynamic range of the sensor. This use of an uncertainty class in distribution space protects against the inevitable unpredictable changes that occur in sensor behavior. The purpose of this research is to examine sensor fusion problems for both linear and nonlinear location data models using statistical decision theory.

The contributions of this research are the delineation of:

- **Robust tests of consistency of data** from different sensors. This confirms that the measurements are actually measuring the same thing. For example, if two position sensors are actually measuring position of two different objects, then the two resulting data points should not be averaged or combined in any way.
- **Robust procedures for combining data** that pass the preliminary consistency tests.

Robustness refers to the statistical effectiveness of the decision rules when the probability distributions of the observation noise and the precise location of the individual sensors are uncertain. This research provides statistical performance bounds.

While range data is a 1D application of multi-sensor fusion, our current research extends the theories to cover the multi-dimensional case. This allows robust testing of multi-dimensional feature vectors for similarities in non-Gaussian noise, and gives a probability of correctness of the result. This represents a significant theoretical advance in mathematical statistics, since no prior probability distributions are assumed. One example of a multi-dimensional application is color comparison. The color of a pixel can be expressed either as R,G,B (where the data is not independent) or in an orthogonal basis. These theories allow us to compare a pixel against another pixel, and give probability of the colors being the same, without assuming in advance that there is a specific probability of that color occurring.

The decision-theoretic formulation of these problems allows us to find minimax decision rules based on a zero-one loss function. Such rules minimize the maximum probability that the absolute error of estimation is greater than an error tolerance ϵ . The zero-one loss function is appropriate for situations where a system works if the decision is correct within a given tolerance, and the system fails if the decision is out of bounds.

This research in robust multi-sensor fusion is based on the theory of robust fixed size confidence sets. Our prior work in this area includes the delineation of the existence, structure, and behavior of:

- optimal, nonrandomized, fixed size confidence intervals based on monotone decision rules;
- robust, nonrandomized, fixed size confidence intervals based on monotone decision rules; and

- the extension of these ideas to both randomized, and nonmonotone procedures.

These results and their applications appear in: [Kamberova and Mintz, 1989], [McKendall, 1990], [Martin and Mintz, 1987], [McKendall and Mintz, 1988], [Zeytinoglu and Mintz, 1984], [Zeytinoglu and Mintz, 1988].

8 Conclusion

As it is seen, the Image Understanding Program in the GRASP laboratory represents a coherent programmatic study of material properties, geometric properties, and motion of objects through visual measurements. Our approach is data driven, where we seek the best explanation of the data via parametric models. The result of this approach is not only compact representation but also a measure of goodness of fit which then can be used for feedback correction depending on the task. We also do not depend only on one measurement, but rather try to combine, in a systematic fashion, several different aspects of shape and material. We are in the process of systematically testing our theories with different parameters of illumination, camera positions, signal/noise ratio and complexity of the scene.

References

- [Bajcsy and Sobh, 1990] Ruzena Bajcsy and Tarek Sobh. *A Framework for Observing A Manipulation Process*. GRASP LAB 216 MS-CIS-90-34, University of Pennsylvania, Philadelphia, PA 19104, June 1990.
- [Bajcsy et al., 1989] R. Bajcsy, S.W. Lee, and A. Leonardis. Image segmentation with detection of highlights and inter-reflections. In *Proceedings of the Image Understanding and Machine Vision*, 1989.
- [Bajcsy et al., 1990] Ruzena Bajcsy, Franc Solina, and Alok Gupta. *Segmentation versus Object Representation - Are they Separable?*, chapter Analysis and Interpretation of Range Images. Springer-Verlag, 1990.
- [Cohen, 1964] J. Cohen. Dependency of the spectral reflectance curves of the munsell color chips. *Psychon. Sci.*, 1:369-370, 1964.
- [Gershon, 1987] R. Gershon. *The Use of Color in Computational Vision*. PhD thesis, University of Toronto, 1987.
- [Gupta and Bajcsy, 1990] Alok Gupta and Ruzena Bajcsy. Part description and segmentation using contour, surface, and volumetric primitives. In Bernd Girod, editor, *Proceedings of the SPIE Conference on Sensing and Reconstruction of 3D Objects and Scenes*, Santa Clara, CA, February 1990.
- [Gupta et al., 1989] Alok Gupta, Luca Bogoni, and Ruzena Bajcsy. Quantitative and qualitative measures for the evaluation of the superquadric models. In *Proceedings of the IEEE Workshop on Interpretation of 3D Scenes.*, 1989.

- [Hildreth, 1984] E.C. Hildreth. Computation underlying the measurement of visual motion. *Artificial Intelligence*, 23, 1984.
- [Ho, 1987] Y. Ho. Performance evaluation and perturbation analysis of discrete event dynamic systems. *IEEE Trans. on Automatic Control*, July 1987.
- [Horn and Schunck, 1981] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185-204, 1981.
- [Judd et al., 1964] D.B. Judd, D.L. MacAdam, and G.W. Wyszecki. Spectral distribution of typical daylight as a function of correlated color temperature. *Journal of the Optical Society of America*, 54, 1964.
- [Kamberova and Mintz, 1989] G. Kamberova and M. Mintz. Robust multi-sensor fusion: a decision-theoretic approach. In *Sensor Fusion II: Human and Machine Strategies*, SPIE Conference Proceedings, pages 1198:192-201, November 1989.
- [Klinker et al., 1988] G.J. Klinker, S.A. Shafer, and T. Kanade. Image segmentation and reflection analysis through color. In *Proceedings of the DARPA Image Understanding Workshop*, 1988.
- [Leonardis et al., 1990] A. Leonardis, A. Gupta, and R. Bajcsy. *Segmentation as the Search for the Best Description of the Image in Terms of Primitives*. Technical Report MS-CIS-90-30, GRASP LAB 215, University of Pennsylvania, Philadelphia, PA, 1990.
- [Martin and Mintz, 1987] K.E. Martin and M. Mintz. *Research on randomized robust confidence procedures*. Technical Report, Department of Systems Engineering, University of Pennsylvania, 1987.
- [McKendall, 1990] R. McKendall. *Minimax Estimation of a Discrete Location Parameter for a Continuous Distribution*. PhD thesis, Department of Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104, May 1990.
- [McKendall and Mintz, 1988] R. McKendall and M. Mintz. Robust fusion of location information. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pages 1239-1255, 1988.
- [Ramadge and Wonham, 1987a] P. J. Ramadge and W. M. Wonham. Modular feedback logidc for discrete event systems. *SIAM Journal of Control and Optimization*, September 1987.
- [Ramadge and Wonham, 1987b] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, January 1987.
- [Shafer, 1985] S.A. Shafer. Using color to separate reflection components. *COLOR Research and Application*, 10(4):210-218, 1985.
- [Solina and Bajcsy, 1990] Franc Solina and Ruzena Bajcsy. Recovery of parametric models from range images: the case for superquadrics with global deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2), February 1990.
- [Tsai and Huang, 1984] R.Y. Tsai and T.S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Trans. Pattern Anal. Machine Intell.*, 6, January 1984.
- [Tsikos, 1989] Constantine J. Tsikos. *Laser Range Imaging System*. GRASP Laboratory, University of Pennsylvania, Philadelphia, PA, May 1989.
- [Waxman and Ullman, 1985] A.M. Waxman and S. Ullman. Surface structure and 3-d motion from image flow: a kinematic analysis. *Intl. Journal of Robotics Research*, 4:72-94, 1985.
- [Zeytinoglu and Mintz, 1984] M. Zeytinoglu and M. Mintz. Optimal fixed sized confidence procedures for a restricted parameter space. *The Annals of Statistics*, 1984.
- [Zeytinoglu and Mintz, 1988] M. Zeytinoglu and M. Mintz. Robust optimal fixed sized confidence procedures for a restricted parameter space. *The Annals of Statistics*, 1988.

Progress Toward An Image Understanding Application Development Environment

Tod S. Levitt, Scott E. Johnston,
Scott Barclay, John W. Dye
Advanced Decision Systems
Mountain View, California

Daryl T. Lawton
Georgia Institute of Technology
Atlanta, Georgia

Abstract

An image understanding (IU) software workstation environment is presented. The environment is aimed at transfer of IU technology into applications development. The C++ environment is based on a hierarchy of core IU objects representing spatial, temporal, inference, interface and storage capabilities. A top level design and implementation to date is shown.

1 Introduction*

This paper presents the design and implementation to date of an image understanding (IU) software application development environment. The core environment provides an integrated set of tools to leverage the development of IU applications and to facilitate transfer of IU, inference and visualization technology from its origins in research laboratories into IU applications. The environment is focused primarily at technology transfer, rather than research and technology development. It provides a development platform and reusable components including a library of image processing and IU routines and data structures, and an integrated set of higher-level reasoning

capabilities such as bayesian networks and logic engines. The design includes:

- An object oriented structure built on the C++ programming language.
- A description of the object representations that are used for the different classes of objects in the environment. Object representations are designed to provide a direct and useful interface to environment capabilities and programming constructs.
- A discussion of user interface, IU routines, inference, database and other capabilities, and how these facilities are integrated with the object representations.

The core set of C++ objects serves as a foundation for the representation of spatial, temporal and symbolic entities central to application development of IU and decision-aiding systems. One reason we have chosen C++ is to facilitate the integration of public domain code, commercial programs and hardware devices. It is intended that application developers will extend the object classes to create objects customized for their application.

The typical application IU system requires signal and/or image processing, geometric and symbolic inferencing capability, an interactive user interface for inspecting and manipulating the processing results, and an associated database for storing the original data and the derived results. Figure 1 shows the roughly hierarchical relationship between these environment components.

* Acknowledgement: This research was funded by the Defense Advanced Research Projects Agency and the U.S. Army Engineer Topographic Laboratories under government contract number DACA76-89-C-0023.

The following sections present our design of these C++ objects and their associated class hierarchy. The environment goals and hardware and

2 Environment Goals and Assumptions

Any IU environment aspires to all of the following goals.

- availability of algorithms
- execution efficiency
- interoperability
- verifiability
- portability
- extensibility
- coding efficiency
- function/data composability
- customizability

Efforts on other IU research and technology transfer environments [Quam, 84, KBVision, 87, Lawton and McConnell, 88, Lawton and Levitt, 89, Waltzman, 90] suggest that the first five goals are of primary importance for environments aimed at development of robust IU applications using well-understood IU technologies, i.e. technology transfer, while the second four are goals associated with rapid prototyping efforts common in IU research and innovative development of IU technology. This effort is focused at the goals that foster technology transfer.

Because of the bias towards technology transfer, and the desire to produce this environment within 2 years, environment component choices have largely been driven by current availability and prevalence of use of hardware and software options. Another driving factor was that as much as possible of the environment should be public domain, so that source code can be provided at minimal cost.

The basic development and user system is a Sun 3, 4 or Sparc workstation with a minimum of 12 megabytes core memory, keyboard and mouse, a color display and at least 300MB magnetic or read/write optical disk. A Vitek image processing acceleration board is under consideration for inclusion in the environment.

The software development environment is the Berkeley Unix 4.2 operating system on a Sun workstation, though compatibility to other Unix implementations and other workstations is maintained where reasonable. We have chosen C++ as the

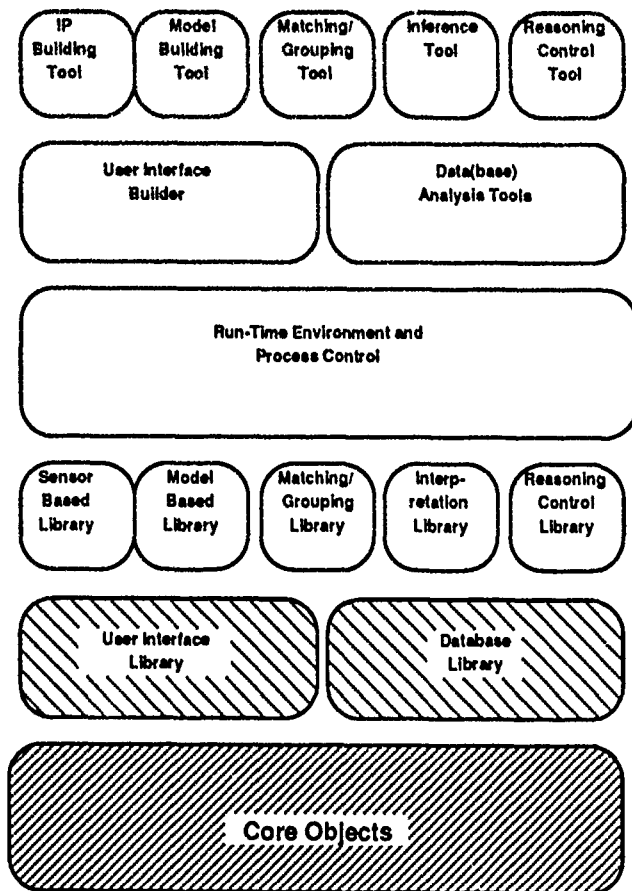


Figure 1. IU Environment Components

software assumptions are described in Section 2. Sections 3-6 describe the core spatio-temporal object classes. Section 7 discusses search and object traversal, including methods supporting perceptual grouping as well as spatial object inference. Section 8 describes user interface objects and methods, and section 9 discusses database classes. Together, these objects comprise the major functional components of the design. Section 10 presents the status of implementation to date. An appendix is provided that lists the top level core object classes.

programming language. This choice is based largely on its efficiency, its relatively good compatibility with C, and its nearterm widespread acceptance in the technology transfer community, i.e. the non-academic IU application development community. We have chosen the Free Software Foundation's Gnu Compiler over AT&T's 2.0 C++ compiler for two reasons. The first is that the Gnu compiler generates faster, more efficient code because it is a true compiler and not just a preprocessor to a C compiler. The other reason is the availability of the compiler source code makes it portable to foreseeable future (Unix) platforms.

X Windows is used for managing displays. The InterViews toolkit from Stanford provides a C++ interface to the X Windows package. IDraw, another Stanford product, provides the interactive graphic window interaction. Chorus, a public domain image processing library from the University of New Mexico, provides both the standard set of image processing functions as well as 2D plotting capabilities. Other public domain software packages being integrated in the basic environment include the CLIPS logic engine and rule-base package, the NCSA 3d display routines, and several neural net packages.

3 Core Spatial and Temporal Class Hierarchy

The class hierarchy is based on the structures developed in PowerVision and View [McConnell et. al., 88, Edelson et.al., 88]. In particular, the basic hierarchy of spatial classes and the concepts of transforms, function concatenation, virtual function wrappers, and programmable database-like search for perceptual grouping were all present in the original PowerVision implementation.

The current design has made strides in uniformity of these structures, cleaned up the relationship between objects and their display methods by associating display methods to the display objects (e.g. windows) rather than the source objects (e.g. a polygon), and has added class structures for coordinates. This design creates fundamental links between the geometric structure implied by coordinates and the programmability of search for

perceptual grouping, as well as the linking together of lower dimensional spatial structures to form higher dimensional structures.

The core objects are organized into four general classes: scalars, collections, containers and coordinates. The scalars are the standard numerics and symbols of C++. Collections are general groupings of objects including arrays, streams, and graphs. Containers are groupings of objects that necessarily have an implied dimensionality and corresponding coordinate systems and imbedding spaces. Containers are inherently spatial: images, curves, solids, voxels, polygons, etc. Coordinates are objects that represent coordinate systems. Local coordinates are objects that are necessarily included within other objects (including other coordinates), while global coordinates can be disembodied.

Containers are designed to wrap around collections, and embed them in a coordinate system. Loosely speaking, we think of the semantic objects in IU systems, such as images, surfaces and volumes, as collections of values associated with coordinate systems. The grouping together in a systematic way of collections with coordinates forms containers. An array of integers is a collection. An array of integers associated with coordinates indicating the context of the array in pixels and centimeters is a container that is of the class Image. Figure 2 shows how containers, coordinates and collections relate to each other, and how they fit into an overall system.

Containers necessarily have coordinate objects and are closely tied to the user interface. The coordinate systems of the containers can map into the display coordinate systems. The display window itself is represented as a container. The necessary projections, translations, rotations, and scaling are implemented by "virtual" containers that wrap around previously instantiated containers and convert them into the appropriately appearing object.

Collections do not have associated coordinate objects, although they can have indices, such as indexes for an array. Collections are closely tied to the underlying devices. For example, a collection can be made to correspond to a device such as an image scanner. The

scanned image becomes an array (one representation of a collection). Efficient access, traversal and transformations are built as methods on collections. Another example is a neighborhood operation like convolution. It can be realized as a collection of data and a method that manages buffers to create fast virtual memory access to the data in the collection.

Tranforms are procedures that operate on containers, coordinates and collections and produce containers, coordinates and collections as output. Although it is possible to represent transforms as containers, providing a pleasing uniformity of data types, it can be semantically confusing to the user. Because technology transfer is a fundamental goal, we have erred on the side of clarity rather than uniformity. So an image is called an image, for example, instead of a function that represents a 2d surface in 3space. We intend to overload class names to permit users both views of appropriate objects.

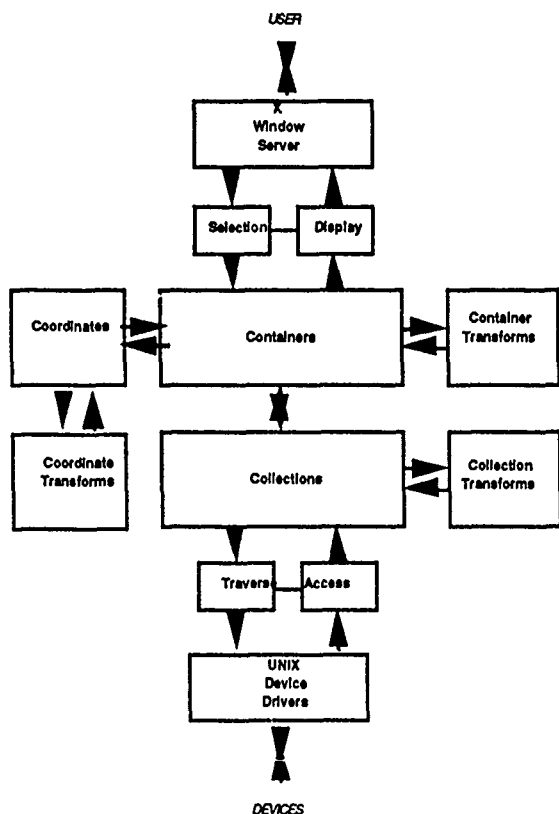


Figure 2. Core Object Relationships

Where it is not confusing, transforms are represented as overloaded constructors

of the class of their output objects. For example, a histogram is a constructor method for the one-dimensional signal that is the output of the histogram transform on an image.

When possible, transforms are defined on containers but implemented on the (coordinate-free) collections to maximize reusability. For example, a one-dimensional smoothing filter can be implemented on an array, then be usable on any linear collection of data, such as an image row, a curve in 3 space, or a specific traversal of the edges of a solid. So the filter can be represented at the more abstract level of the container hierarchy as a method on a curveNd (i.e. a one-dimensional curve in N space), enabling polymorphism.

The next three sections describe the collection, coordinate and container objects in detail. This is followed by a description of how containers and collection objects are efficiently traversed, accessed and searched.

4 Collection Classes

Three classes of collection objects are planned: Stream, Graph, and Array. They can be characterized by the style of traversing and accessing the collection. Streams are traversed in a sequential manner, where the next access is restricted to the neighbor in a single forward direction. Graphs are traversed in a linked manner, where the next access is restricted to nearest neighbors in any direction. Arrays are traversed in a random manner, where the next access is unrestricted.

The collection class hierarchy can be extended to wrap a stream, graph, or array around external sources. A character stream can be wrapped around a serial port. An array can be wrapped around a frame buffer. A graph can be wrapped around a Connection Machine or Transputer topology.

Graphs are the general case of a linked data structure. A tree is a subclass of graph, and a list is a subclass of tree. This class hierarchy allows lists and trees to be manipulated by graph traversing routines, e.g., "WalkDepthFirst". Graphs can store heterogeneous collections of objects. Graphs are implemented with separate node and arc

objects. At this time we do not plan to support special subclasses for specific types.

The basic methods for a collection are as follows:

- constructors: creation and conversion routines
- destructors: memory, process, and device deallocation routines
- printers: ASCII printing routines
- traversers: universal location generation routines
- searchers: selective location generation routines
- accessors: value access routines.

"Traversers" are methods for traversing the object, visiting each member object or element in turn. Each traversal of an object has an associated current location. Traversers accept a function pointer

argument and apply the function at (a neighborhood of) each location.

"Searchers" are incremental, partial traversal methods. A search routine operates at the current location and chooses which location(s) to visit next. A search routine accepts two function pointers, applying one to the current location (neighborhood) and the other to choose the next location(s).

"Accessors" are methods for accessing the member object stored at the current location in the collection object. Access can be by value, or by reference to allow for overwriting.

5 Container Objects

Container objects represent an S-dimensional containment of objects in R-space.

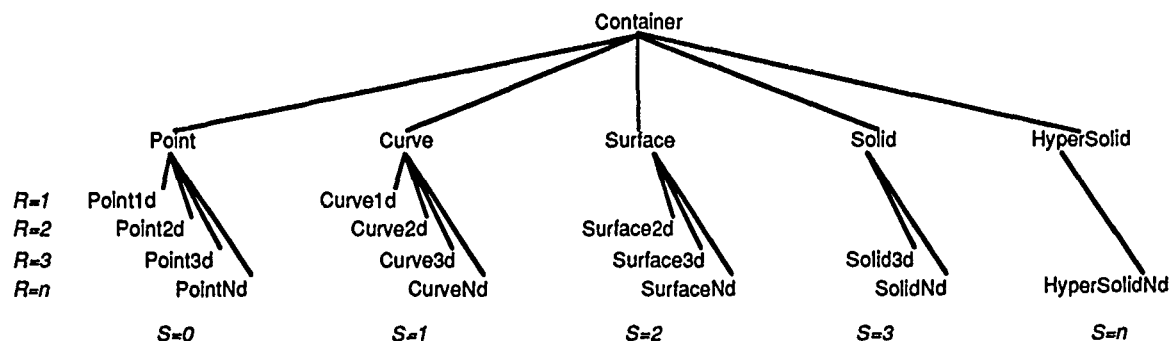


Figure 3. Top Level of Container Class Hierarchy

Figure 3 shows only the top levels of the container inheritance hierarchy. Additional subclasses are derived so that the leaf node classes of the hierarchy are realizations of more familiar spatio-temporal data structures and procedures. A signal is subclass of a one-dimensional container in 1-space. An image is a subclass of a two-dimensional container in 2-space. A volumetric representation is a three-dimensional container in 3-space. Constraints for a linear programming problem can be viewed as an N-dimensional container in M-space.

The container subclasses are effectively parameterized by the dimensionality of the container's topology, and the dimensions of the imbedding space. A 2d curve is a one-dimensional container in 2-space. A 3d

curve is a one-dimensional container in 3-space. A polygonal region is a two-dimensional container in 2-space. A polygon3d is a two-dimensional container in 3-space. A terrain elevation map is a two-dimensional valued container in 2-space.

Specific classes of container are represented in multiple ways. For example, a three-dimensional container in 3-space is a solid, and a solid can be represented functionally ($X^2 + Y^2 + Z^2 \leq 1$), volumetrically (via voxels or octree), or by a surface model.

The cross-product of the S-dimensionality of the containers and the R-dimensionality of the embedding space is represented by a class hierarchy where the top-level branching is container dimensionality and the lower-level branching is embedded space

dimensionality. Point, Curve, Surface, Solid, and HyperSolid are the superclasses, and their subclasses correspond to the space the container is in.

To achieve efficiency, 0d, 1d, 2d, and 3d containers are implemented as special-cases, and Nd containers are handled in a general fashion. In the same manner, containers embedded in 1d, 2d, and 3d spaces are implemented as special cases, and embedding in Nd is handled in a general fashion. Beneath each branch of the container hierarchy are three subclasses that reflect increasingly general ways of representing a container:

- 1- Constant Containers
- 2- Valued Containers
- 3- Connected Containers
- 4- Aggregate Containers

Constant containers describe the shape of a container without representing its values, or "contents". The shape is defined to be its geometric representation in Nspace, without values necessarily being defined at locations of the shape. Because a shape is geometric, it usually has a boundary that we call its "shape boundary" to distinguish it from other uses of the term. For example, a solid cylinder in 3space has a solid cylinder as its shape, and a hollow cylinder as its boundary shape.

We can represent a force field acting on the solid cylinder by associating the appropriate local magnitude and direction of the force field with each point of the shape. This is an instance of a valued container. Valued containers have a shape description and a content mechanism, whereby values such as scalars or more complex objects can be associated or stored with each shape location.

Connected containers group other containers, relating them with a series of coordinate transforms or other relations such as adjacency or attachment, and merging them into a single connected entity. A CAD model of a single car built from surface facets is a connected container. A smoothing pyramid is a connected container, where image objects are related (connected) by the order of the smoothing and sub-sampling operations that created them.

Aggregate containers group a disjoint set of containers into a single entity. The set of CAD models of all cars manufactured at a particular plant is an aggregate container.

The basic set of methods for a container are the following:

- constructors: creation and conversion routines
- destructors: memory/process/device deallocation routines
- printers: ASCII printing routines
- traversers: universal location generation routines
- searchers: selective location generation routines
- accessors: value access routines
- draw: draw representation of self in an X Window
- display: add self to display list
- inside: predicate to determine if point is inside boundary.

"Traverser", "Searcher" and "Accessor" methods typically window through to an underlying collection. The current position of a container traversal is in effect a current position of the underlying collection traversal, and the mechanism for accessing the data in the container is the same as the mechanism for accessing data in the underlying container or collection.

"Display", "Draw" and "Inside" are methods for realizing the user interface. The display method queues the object for display in an X Window by placing the object on the display list of the window. Then the window object takes care of determining the necessary parameters to call the object's draw method. An object's draw method produces pixel values that are a representation of itself and maps them into a window display, or any container of type Surface2d. The inside method is used by the window object for each object on the display list to determine if a particular mouse click has fallen within its bounds. Only containers and coordinates can be displayed in the 2D and 3D object windows, as coordinates are required to relate to the window display. It is possible to display collections in structured text or graph browsing windows

6 Coordinate Objects

Coordinate objects represent coordinate systems. A coordinate has a corresponding type that is one of cartesian, polar, cylindrical, spherical, quaternionic, or shape. Shape means the coordinate system is defined in terms of distinguished points in a container, like attachment points, or the ends of axes of sub-objects. A local coordinate is necessarily contained in another object such as a container or another coordinate. A disembodied coordinate is defined to be the subclass of global coordinate. Coordinates have methods that act as transformations between other coordinate systems. A coordinate records its transformations between other coordinates, unless these transformations are explicitly deallocated.

There are two subclasses: global and local. Global coordinates can occur disembodied, i.e. without being contained in or referencing other objects. They can be transformed and copied by any coordinate constructor to mix in when constructing a local-coordinate defined for a container or other global or local coordinate. This "places" the container in the global coordinate system. The global coordinate remembers the containers that were constructed with it.

A local-coordinate can represent the imbedding space of the container, or other ego-centered coordinate systems. An object can have multiple local coordinates. The base-coordinate is a distinguished local coordinate. It is defined to be the first local-coordinate associated to a container or other coordinate. The base-coordinate is instantiated by the container constructor. It can be specified by the caller of the constructor method. The base-coordinate is guaranteed to have transforms associated to all other local-coordinates of that container or coordinate. It is intended, although not required, that the base-coordinate correspond to the natural traversal of an associated collection of values. For example, a raster image is a Surface2d container whose values are in an Array2d (collection). Its natural base-coordinate is the cartesian coordinate with origin at array index (0,0), one axis in the row direction and another corresponding to columns. For a pyramid, a

natural base-coordinate is similar, but includes a third axis in the multi-resolution direction.

Coordinates all contain the following methods. Note that when local and global are not explicitly called out, either applies. For example, the transform method can relate locals to locals, locals to globals or globals to globals.

type: returns a mathematical type (e.g. cartesian) or the type "shape".

origin: returns a point

dimensions: returns list of dimensions

units: returns list of named units per dimension

minextents: returns list of minimum extents per dimension

maxextents: returns list of maximum extents per dimension

convert: inputs a type that is not "shape" and creates versions of its local-coordinates expressed in that type (e.g. cartesian to polar conversion)

list-transforms: returns the list of transforms known between itself and other coordinates

transform: inputs another coordinate with a known transform to itself, and a third coordinate with a known transform between it and the second coordinate; returns a transform between itself and the third coordinate.

propagate-transform: inputs a coordinate with known transform between itself and the coordinate, and returns the list of transforms between all its local-coordinates and the input coordinate.

7 Object Traversal and Search

From the user's point of view, containers get traversed or searched. From the workstation environment's point of view, containers are pointers to collections that get traversed or searched. Both traversal

and search can be thought of as routines consisting of the cyclic applications of three functions: move, access, and apply. In the case of traversal every value in the underlying collections is necessarily visited, so the function for moving, or choosing the next location(s) in a container's shape, is known before traversal is invoked.

In search all locations/values are not necessarily visited. The move function must be passed by reference to the search method. The apply function is invoked on the appropriate neighborhood at each visited location for both traversal and search methods.

Signal and image processing functions traverse their contents to enable extraction of higher-level interpretations. These routines need to quickly iterate across their N-dimensional data sources, with efficient access to a local neighborhood ranging in size from 1 to M units in any dimension.

Traversal is intended to provide the support for a programming style whereby the application developer codes the operation to be done at each point in the traversal, and leaves it up to some other mechanism to slide this operation around the container. This requires two things: an underlying mechanism for efficient traversing (tied to an efficient accessing scheme) and a programmer interface.

When a programmer is presented with an efficient source of neighborhood data, it is convenient to string together a series of smaller functions to do the work of a more complex function. However, the programmer is typically forced to write the complex function out flat, inline in one function, to avoid the overhead of piping data between functions. The IU workstation environment provides support to concatenate existing low-level operations without incurring extra overhead.

This can be done by constructing a library of neighborhood operations that describe what is done on one neighborhood, but contain no mechanism for iteration. Examples are convolution kernels, median filtering, and basic arithmetic and logical manipulations of Nd data. Neighborhood operations that maintain a state are implemented in this model by saving the permanent state in static and/or global variables for later retrieval.

This makes the process of writing more complex neighborhood operations into one of concatenating the series of operations into a single neighborhood operation. A specific neighborhood function is then inserted in the middle of a looping mechanism that is capable of traversing the container, and supplying the neighborhoods of data to the operator.

Object access is streamlined with a caching mechanism that makes a local neighborhood available to the C++ program in an internal C++ data structure. The mechanism is program-controlled, in that the program decides when to initialize it and when to refresh its contents. Because the cache is represented as a standard C++ data structure, either an array or a nested array of pointers to arrays, the efficiency of data access within the cache is identical to array-based data access.

A neighborhood cache is useful for representing windowing operations on imagery. The input object is an image, the cache is an array of pointers to linear arrays; as the window slides across the image, the cache is refreshed by updating the pointers.

For point transformations the caching mechanism is useful in order to reduce the overhead of row access. The cache is defined to be a single array equal in length to the image row, and it is refreshed after each row is processed. The processing of the row is done with a tight for-loop, with entirely in-memory data access.

The ability to compose functions without creating intermediate data structures yields the ability to display the results of experiments with a minimum of typing on the part of the programmer (e.g. not creating named functions as above) and with a great saving of memory and memory management overhead. In Powervision functions created this way were called "pixel-mapping-functions". Because individual objects know how to display themselves, the pixel-mapping-function capability is re-created with a wrapper that says: compose the following functions (passed by reference) on this input data, and add the display method for the result of the last function at the end. Ordinarily, the final result is saved, because it is often the input to a next stage of processing.

For example, to apply a convolution to only the pixels in an image defined by a mask, a search method is applied to the image object, where the search method run length encodes the mask and accesses the "on" pixels only. The search method is composed with the convolution to feed only the relevant neighborhoods to the convolution kernel.

Mapping functions are implemented by subclasses of their respective containers. There are four basic types of mapping functions, and so four basic mapping function class extensions:

- 1) coordinate transformation of container locations
- 2) look-up-table applied to container values
- 3) arbitrary expression applied to container values
- 4) arbitrary expression applied to container locations.

8 User Interface

The user-interface supports the direct manipulation and inspection of all entities in the vision environment through a windows-menu-and-mouse interface. The user interface is based on X Windows, a network window system, and InterViews, a C++ package that defines basic X Windows objects. The user interface must be capable of displaying a list of containers to an X window, and mapping mouse clicks to specific objects in the display list. The following presents the display list object, window types and addresses issues in imagery display and mouse protocol.

The display list is an object that keeps track of what set of objects is currently being displayed in a window. There is a single display list associated with each window. The display list is implemented as a connected container of 2d surfaces. The connected container groups two-dimensional points, curves, and surfaces, interrelating them with coordinate transforms and other programmed relations. Each container stored in the display list knows how to redisplay itself, and knows how to determine if a given point is inside or outside its 2d shape boundary or whether a given rectangle overlaps its shape boundary.

The window system supports overlapping windows, as well as neatly tiled windows, useful for applications once they reach a certain level of maturity.

While in overlapping mode, each window can be resized, repositioned on the screen, collapsed down to an icon, and expanded back to its original size and shape. When overlapping mode is disabled (tiling mode), the size and shape of each window is predetermined (or tightly controlled), and the opening and closing of windows is directed by the application.

Each window is associated with a display type that governs the type of information that can be shown within the display region. The supported window types include:

- 2D Object Display Window
- 3D Object Display Window
- 2D Plotting Window
- 3D Plotting Window
- Directed Graph Browser Window
- Structured Text Browser Window
- Dialog Box Window.

The display of objects to windows is object-oriented, in that each entity within the vision environment knows how to display (or present) itself to a window of a specific type.

Most graphics that overlay images occupy a small area compared to the image size. An example is the overlay of linear features, such as roads or rivers, on an image.

When the image is drawn, it is from one container object. Each window is associated with a display type that governs the type of information that can be shown within the display region. The linear features are assumed to be a second container object with coordinates that overlap the image. The problem is to allow the user to select and unselect the display of the graphic overlays without redrawing the whole screen just to refresh the small area under the graphic overlays.

The approach is to create a third object that has the same container (i.e. "shape") information as the graphics, but uses the values from the image collection. Refreshing the screen is accomplished by requiring the appropriate set of graphic objects to refresh themselves. This capability is called a "sprite" object in the object-oriented imagery display literature.

9 Databases

Database functions include:

- simple persistent storage of images, objects, functions and other data,
- flexible conditional queries to retrieve or compute instances of objects,
- structural ordering of the object instances to provide fast, efficient access to the objects,
- consistent convention for accessing a variety of different objects with a minimum of coding effort, and
- extensibility to support group data sharing on different physical databases.

The database manager is organized in a client-server model and consists of two components, the database interface and the database server. The database interface (or client) provides the interface to the database from any other programs. This interface is provided as methods that the other objects may invoke. Such methods include: insert, delete, save, find, etc.

The database server manages the storage and access to items assigned to the database including the allocation and deallocation of space. This includes creation, documentation, modification, access, and deletion of user objects (images, features, etc.) and programming objects (functions, documentation, numbers, characters, etc.).

Databases traditionally provide a shared access that prevents two users from interfering with each other when accessing the same object and provide a transaction system to ensure the integrity of each interaction with the database. These aspects of a database are not as important

for the IU environment and are not discussed further here.

The approach to the database design is in two levels. The basic level provides for the storage of objects, groups of objects and indices as files for management by the operating system. These files can be stored and read directly from the program or under interactive user control.

The next level provides interface to database management systems from commercial products. The current plan is to develop a generic SQL interface for the class hierarchy. This allows interaction with standard relational database system (e.g. Sybase, Ingress, Oracle). Interface to new object oriented databases (e.g., Ontologies' Ontos) is a future possibility. Hooks are provided to build additional structures for efficient access, such as quadrees.

The database is integrated into the core workstation software in several ways. The primary access to the database is through the C++ language. The user (developer) takes advantage of the database through the core object structure.

Our approach to the database uses the strong typing feature of C++ by allowing the database objects to be incorporated directly in the object hierarchy transparently. That is, the objects are compiled directly into the program (with strong type checking) and are stored in the database by invoking the persistence attribute. All imagery, imagery objects, functions, production rules, reasoning structures, etc. are expected to be stored in the database and accessed through the same C++ program interface. A set of user interface procedures that form a front end to the objects stored in the database are provided for the developer.

The Sybase relational database system provides the basic relational database capability. An object oriented view of the database is provided by a set of object oriented procedures used as a front end to the relational database.

This approach assumes that the structure of objects stored in the database is known to the compiler; in fact, the object storage mechanism is compiled and linked with the application program. It also assumes that the persistent objects have a standard set of methods for storing, retrieving, inserting, ordering, etc. These methods constitute the interface to the database and must be

chosen carefully to allow replacement of the database structure at a later date.

The index or ordering information on objects in the database is provided by special objects that have the appropriate structures. Any group of objects may be ordered using this object type by creating an appropriate indexing object. These indexing objects include: binary trees, quad trees, oct trees, hash tables, etc. The indexing objects access the ordered objects by providing an offset into a table storing the data for these objects.

Another function of the data base is to store information derived from the objects. These are arbitrary sized objects and may be retrieved by a variety of attributes. The attributes of these objects are defined by methods on the objects. These methods/attributes may be precomputed and stored in an indexing list or they may be computed when the query is made.

10 Current Status

The current environment status represents 5 months of design and 3 of implementation on a 27 month effort. The Chorus image processing package is not yet available as of the writing of this paper. Hence, implementation results focus on basic user interface and database capabilities.

Recall that InterViews and IDraw are public domain object oriented user interface toolkits built on top of X Windows. To date, ADS has extended the graphical object hierarchy of InterViews and IDraw in two ways: the addition of images and of Bayes nets.

Within IDraw, images are first class objects. The user can put an image object into the drawing by clicking with the mouse and pulling out a rubber rectangle to define the outline of the image. The system presents the user with a menu of files, and when the image file has been chosen, inserts the image into the designated rectangle in the drawing, clipping the image if necessary.

Once an image object is defined and displayed, the user can perform a variety of drawing operations on it, as with any drawing object. Images can be moved, scaled, stretched in width or height, or rotated. Arbitrary image warping is currently being implemented. In addition

the user can draw any kind of graphical object on top of the image, and then group the object with the image, allowing drawing operations to be performed on both objects simultaneously. For example, the user can draw a colored polygon over a region of interest on an image, then group the polygon with the image into a composite object, then scale and rotate the composite object. The polygon still covers the same area of interest on the image. These capabilities are shown in figures 4. (To save space, some photos have been cropped so that the full computer screen is not shown.)

Rather than detailing each additional capability, we show our current state of implementation through two interactive processing scenarios. They demonstrate the benefits of an integrated object hierarchy, the use of images as first class objects, uniform representation of display and interactive object manipulation, and seamless access to remote processes.

The first application is diagnosis of arthritis from evidence extracted from a hand xray pictured in figure 5. Nodes and links are included as graphical objects. Graphically accessible methods are associated to form, in this example, a Bayes net object. Evidence can be acquired from images by measurement, and the evidence propagated through the Bayes net. Probabilities can be graphically inspected. For example, given a Bayes net that draws inferences about a disease condition of arthritic hands called periarticular demineralization, it is possible to take measurements on an xray of a hand in order to obtain evidence for the Bayesian network.

As illustrated in figure 5, the user loads the xray as an image object, draws a line down the middle of one of the finger bones (phalanges) and asks for a plot of the intensity values under the line by selecting "Profile" from a menu. The plot is shown in a window. A measure is taken of the relative density between the ends of the phalanx and the average density along the axis. This measure is added to the Bayes net as evidence by selecting the image, line and relevant Bayes node and selecting "Add Evidence" from the list of Bayes net menu options. The impact of the evidence at any point in the net can be seen by selecting

the desired node and the menu choice "Show Belief". It is displayed as a probability histogram over the possible hypotheses at the node. In figure 5 these hypotheses are "demineralization" and "normal".

The second application scenario involves interactively querying a digital terrain database stored in Sybase. Figure 6 shows

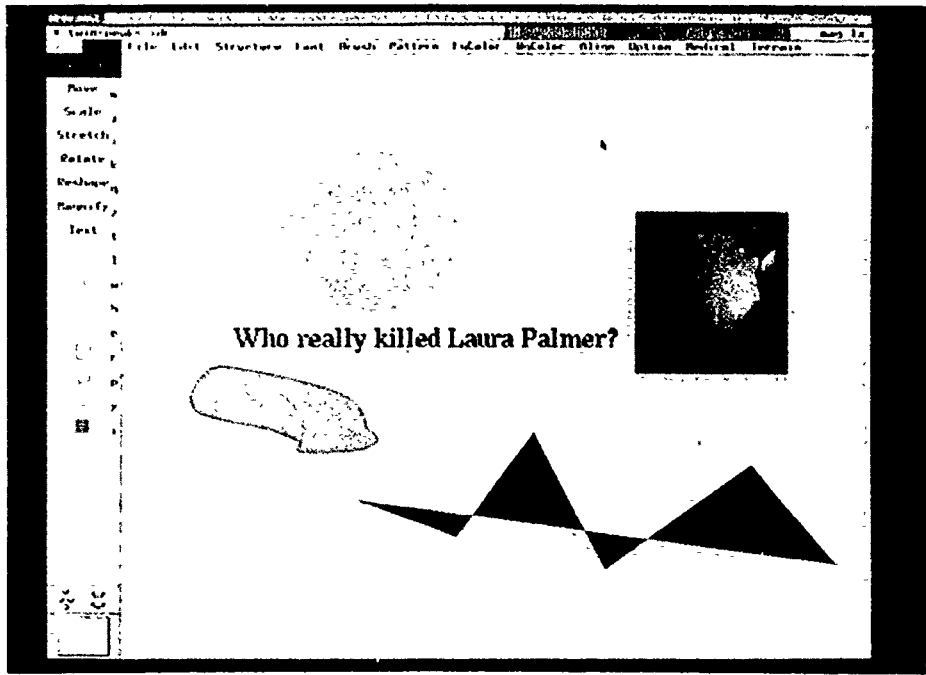


Figure 4. Image Display, Manipulation and Graphic Overlay

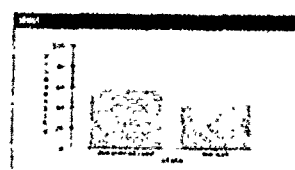
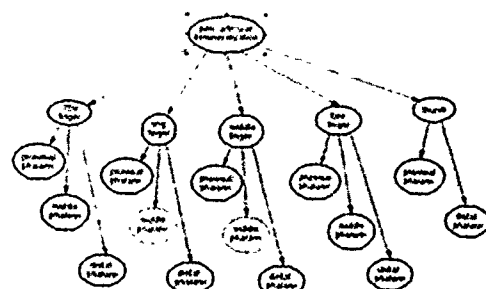
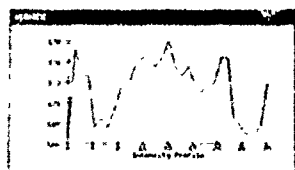
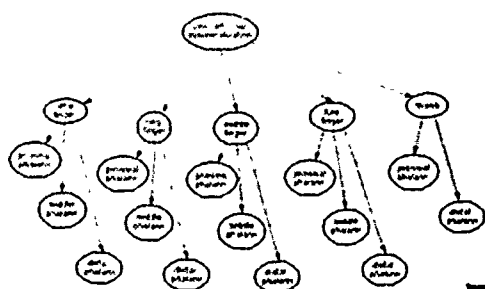
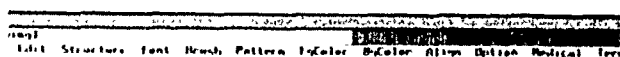
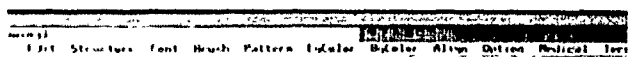


Figure 5. Integrated Image Feature Extraction and Bayesian Inference

seamless interaction with an external process through a graphical interface. The user brings in an image of a map that is registered with the digital database. A region of interest is selected by drawing an ellipse on the map. Selecting the appropriate terrain layers and the

"Retrieve" option from menus, a message is sent to Sybase, generating an SQL query. In this case, the database is populated with data on offshore oil wells, so a popup window of the wells in the region is displayed when the query results are returned.

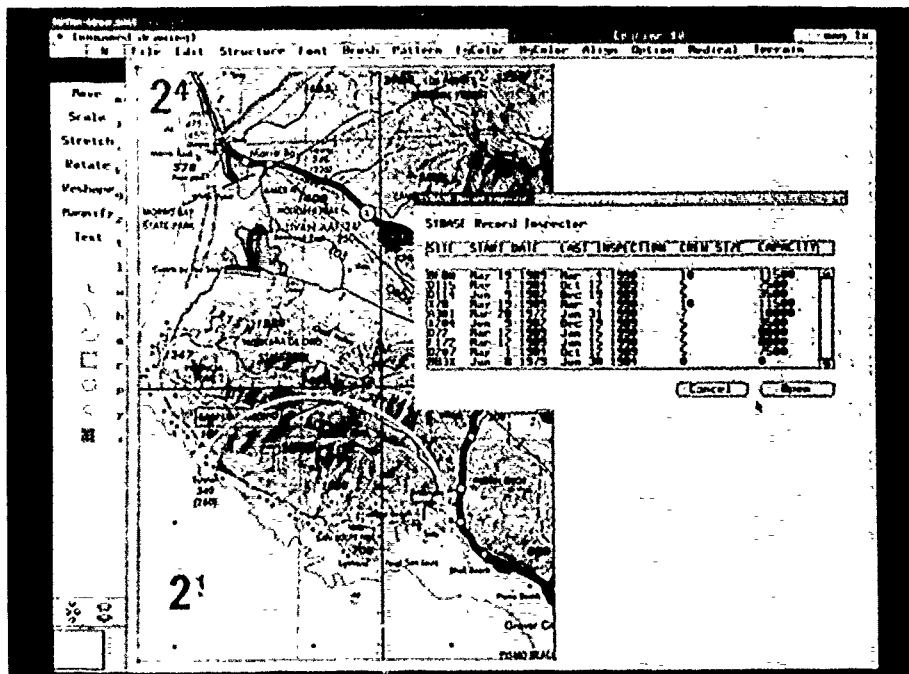


Figure 6. Graphical Sybase Interaction

References

- [Edelson et.al., 88] Edelson, D., J. Dye, T. Esselman, M. Black, and C. McConnell, "VIEW Programmer's Manual", Advanced Decision Systems, Mountain View, California, June 1988.
- [KBVision, 87] "KBVision Programmer's Manual", Amerinex Artificial Intelligence, Amherst, Mass., 1987.
- [Lawton and Levitt, 89] Lawton D.T. and T.S. Levitt, "Knowledge-Based Vision for Terrestrial Robots", Proc. DARPA IU Workshop, Morgan Kauffman, San Mateo, California, May, 1989. pp 128-133.
- [Lawton and McConnell, 88] Lawton, D.T. and C.C. McConnell, "Image Understanding Environments", Proc. IEEE, Vol. 76, No. 8, August, 1988, pp. 1036-1050.
- [McConnell et. al., 88] McConnell, C., K.Riley, and D. Lawton, "Powervision Manual", Advanced Decision Systems, Mountain View, California, 1988.
- [Quam, 84] Quam, L., "The Image Calc Vision System", Stanford Research Institute, Menlo Park, California, 1984.
- [Waltzman, 90] Unpublished notes at the Image Understanding Environment Requirements Meeting, Hughes AI Laboratory, Malibu, California, May, 1990.

Appendix A: Top-Level Class Hierarchy

Container

Point

- Point1d
- Point2d
- Point3d
- PointNd

Curve

- Curve1d
 - Signal
- Curve2d
 - SimpleCurve2d
 - PointCurve2d
 - EdgeCurve2d
 - PolynomialCurve2d
 - SplineCurve2d
 - BezierCurve2d

Curve3d

- SimpleCurve3d
 - PointCurve3d
 - EdgeCurve3d
 - PolynomialCurve3d
 - SplineCurve3d
 - BezierCurve3d

CurveNd

- SimpleCurveNd
 - PointCurveNd
 - EdgeCurveNd
 - PolynomialCurveNd
 - SplineCurveNd
 - BezierCurveNd

Surface

- Surface2d
 - SimpleSurface2d
 - ConstantSurface2d
 - Box
 - Polygon
 - Parallelogram
 - RLE
 - ValuedSurface2d
 - Image
 - PolygonImage
 - WarpedImage
 - TiltedImage
 - RLE_Image
 - ConnectedSurface2d
 - AggregateSurface2d
- Surface3d
 - SimpleSurface3d
 - ConstantSurface3d

- ValuedSurface3d
- ConnectedSurface3d
- AggregateSurface3d
- SurfaceNd
 - SimpleSurfaceNd
 - ConstantSurfaceNd
 - ValuedSurfaceNd
 - ConnectedSurfaceNd
 - AggregateSurfaceNd

Solid

- Solid3d
 - SimpleSolid3d
 - ConstantSolid3d
 - GeneralizedCylinder
 - CSG's
 - ValuedSolid3d
 - Space
 - BoundedSpace
 - ConnectedSolid3d
 - AggregateSolid3d
- SolidNd
 - SimpleSolidNd
 - ConstantSolidNd
 - ValuedSolidNd
 - ConnectedSolidNd
 - AggregateSolidNd

HyperSolid

- HyperSolidNd
 - SimpleHyperSolidNd
 - ConstantHyperSolidNd
 - ValuedHyperSolidNd
 - HyperSpace
 - BoundedHyperSpace
- ConnectedHyperSolidNd
- AggregateHyperSolidNd

Coordinate

- Global
- Local
- Base

Collection

Array

- ByteArray
 - Array2d
 - ByteArray2d
 - Array3d
 - ByteArray3d
 - ArrayNd
 - ByteArrayNd

Stream

- ByteStream

Stream2d
 ByteStream2d
Stream3d
 ByteStream3d
StreamNd
 ByteStreamNd

Graph
 Tree
 List

Record

Image Understanding Research at Brown University

D. B. Cooper, T. L. Dean, W. A. Wolovich

Brown University,
Providence, RI 02912

Abstract

This is a new interdisciplinary center in the DARPA Image Understanding program. Since our group effort has just recently begun, the proceedings space available has been devoted to four papers describing some of the key concepts that are being explored. These four papers are identified in the reference section of this paper. Due to spatial limitations, this paper, the principal investigators' program overview, is brief and only touches on the primary directions in our program.

1 Introduction

The proposed work is a multidisciplinary effort consisting of basic research and a number of demonstration projects in image understanding, with an emphasis on service robots.¹ The purpose of the demonstration projects is to act as a focus for much of the research, and to produce subsystems, significant portions of which should be exploitable for military applications. This will involve our focusing on computational and accuracy considerations. Since the systems must deal with considerable complexity, planning capability is of importance here. In general, more than one type of sensing will be necessary. We plan to use passive stereo and continuous-contact force/torque sensing in our initial investigations. Some use will be made of active rangefinding—most likely based on the use of structured light. The demonstrations will involve topics central to image understanding for mobile service robots in general, and therefore will be applicable to robots for servicing military vehicles, loading munitions, warehousing, scraping paint on shipboard, scrubbing surfaces, etc.. The image understanding will be equally useful in indoor and outdoor applications, although the experiments will be primarily indoor.

2 Experimental Environment

To focus the basic research on topics important to practical applications and for the purpose of testing the al-

gorithms developed, our experimental environment will be that of object recognition, object position estimation, scene understanding, and navigation in laboratory environments. The tasks involved are those that might be described as the functions of a repair person's assistant. Ideally, the robot should be able to respond to command tasks such as: *Go to the next laboratory and bring back a logic analyzer that is sitting on one of the worktables.* For this purpose, the robot must find its way to the next laboratory, which is an unstructured man-made environment, recognize worktables and navigate to them, recognize a logic analyzer on one of the worktables, and estimate the position of the analyzer and the free space around the analyzer in order to determine how to grasp the analyzer and then move it. The robot should be able to grasp and move small objects. Moving large objects would require the coordinated use of two sizable arms, which is not in our projected experimental facility.

What is the complexity of the scenes that we expect to deal with? There are a few classes of large objects in the environment. They consist of: *immovable* walls, room partitions, and large storage cabinets; *rarely moved* desks, worktables, bookcases, and file cabinets; *frequently moved* chairs. There will also be moving people, and cartons and other clutter that are on moving platforms or on the floors but which are frequently moved. Objects that the robot must recognize and estimate in order to manipulate them are largely small tools, parts, and instruments. They will usually be on worktables, desks, or the floor. Occasionally, they will be on mobile stands.

Because there are a number of disciplines that are being brought together in this project, the most practical procedure appears to be to pursue two developments in parallel. One is primarily vision and force/torque sensing for high and low resolution scene and object recognition and estimation using planning. This will be model based. The experimental facility will be a robot arm with a force/torque sensor and a black and white ccd camera, and a pair of color cameras that can pan, tilt, and zoom on a stationary base.

The second development will be vision, tactile, sonar, and proximity sensing for low resolution scene estimation for navigation. This research will be carried out using a mobile platform, and will emphasize planning for real-time sensor fusion and navigation. The mobile

¹Future investigations in the areas described will be supported, in part, by the NSF and DARPA under Grant No. IRI-8905436.

platform will have some on-board computing, but we anticipate the need for additional computing resources and hence are looking into communication hardware to off-load some of the computing. In addition to permitting research on vision and tactile sensing, and research on planning, to proceed simultaneously, the two developments will permit assessing the relative merits of using fewer types of sensing with higher resolution, more-complex scene models, versus more types of sensing and lower resolution scene models with more-complex reasoning. A complete system may find that one approach is better for some tasks, and the other approach is better for other tasks.

3 Major Research Topics

The demonstration projects are important because they focus a significant portion of our research effort on that image understanding that is central to the realization of practical service robots. We feel that the repair person's assistant creates a useful laboratory environment because it starts at a practical technology level, and problems of increasing difficulty can be tackled and incorporated into increasingly sophisticated systems. The Denning surveillance robot and the TRC helpmate robot, for transporting food and supplies to locations in a hospital, are examples of presently useful technology.

Basic research will be covered that is generally applicable to indoor and outdoor scenes, as well as to the demonstration projects. The systems aspects of the projects will force considerations that would not arise were some of the topics to be considered alone. Among the major topics that will be covered are the following.

1. 3D geometric modeling with emphasis on high-degree polynomials in x, y, z , for surfaces or groups of surfaces, and algebraic representations for nonplanar curves or groups of curves. These curves are represented as intersections of polynomial surfaces. This is both for objects that are to be recognized or manipulated and for scenes. This work can be viewed as extensions of the use of collections of planar and quadric patches for modeling 3D surfaces or 2D curves, or using collections of high curvature points as significant features for object representation. Since our higher degree polynomials can represent larger patches of surfaces, which may be disconnected, by a single analytic function, they can capture more significant structure of objects, thus providing greater discriminatory power that is important for object recognition and position estimation when the number of possible objects is large and clutter and occlusion are prevalent. They also appear to have computational advantages. Objects of interest will often belong to large classes, e.g., the class of chairs. Since the number of different shapes in this class can be large, we will model the geometric variability probabilistically. Markov Random Fields show promise for this purpose. This is true for other highly variable surfaces as well, such as human figures or creased or crumpled paper.

2. 3D surface estimation, volume occupancy determination, object recognition, and object location and orientation estimation. These are all difficult problems with respect to computational cost and inference accuracy for

realistically complex situations. Among the approaches we will be taking in dealing with these situations are those in [Taubin and Cooper, 1990], [Cooper *et al.*, 1990], and [Dean *et al.*, 1990]. For 3D surface or curve or 2D curve recognition and position estimation from data, [Taubin and Cooper, 1990] introduces: approximations that permit computationally modest generalized eigen methods for fitting algebraic functions to data; special polynomials for use as features in a large object data base; geometric invariants. The special polynomials are of two types: one is a single polynomial that represents a group of primitive surfaces, e.g., three planes, or a plane and a 3rd degree surface; the other polynomial is what we call an "interest region". It is a region where the polynomial changes little for small changes in the region used, and is such that a 3rd or 4th degree polynomial fits the data well, but a polynomial of lower degree does not. In [Cooper, *et al.*, 1990], the problem of 3D surface estimation, recognition, and segmentation from a sequence of images is put into a general Bayesian framework. This appears to be able to handle essentially all situations from highly variable surfaces through segmentation into objects. In [Dean, *et al.*, 1990], we present an approach to building planning and control systems that combines sensor fusion and sequential decision making for active perception. The approach is based on Bayesian decision theory, and involves encoding the underlying planning and control problem in terms of a compact probabilistic model for which evaluation is well understood. We illustrate our approach using a robotics problem that requires spatial and temporal reasoning under uncertainty and time pressure. We use estimates for the computational cost of evaluating the probabilistic model to justify representational tradeoffs required for practical application.

3. Continuous-contact tactile (force/torque) sensing. Our primary objective, with respect to force/torque sensing, is to determine how touch can be used for object recognition via force-contact motion control along the surface of an unknown object. Since force information employs only "local" information, in that it involves only the contacted object, and not the environment, far less information need be processed (when compared to vision) in order to identify a contacted object. We have developed a new technique, termed "dual-drive" control, for moving the end-effector of a robot along a surface, without explicit knowledge of the surface, in order to obtain trajectory information which can subsequently be used to identify the object. This work is detailed in [Wolovich, 1990].

4. Sensor fusion. This involves fusing models and data from multiple images for passive stereo, tactile data, and data from active sensing. The active sensing will be structured light and perhaps others. Our approach is to model the structural mutual dependencies among the data sets produced by the various sensors, and then treat recognition or position estimation as decision theoretic problems based on these mutually coupled data sets. We have developed machinery for doing this down at the raw data level. Examples of this are global inferences about 3D objects from patches of range data [Bolle and

Cooper, 1986] and 3D surface estimation from a sequence of images [Hung *et al.*, 1990]. The challenge in the proposed research will be to use these tools we developed for limited domains, and apply them to data from this broad range of sensors. Other tools will be brought to bear as well [Dean *et al.*, 1990], and additional ones will have to be developed.

5. Planning for image understanding occurring in task implementation. This involves determining the appropriate level of sensed information for the task at hand; choice of appropriate sensors; determination of the most advantageous sensor position; grasping and manipulating objects in a cluttered environment, and navigation in a complex environment, guided by information obtained from sensors. Bayesian decision theoretic techniques play a role here, and useful approaches and results have been obtained in recent years by us and others.

6. Learning Environmental Geometry. Learning objects, classes of objects, and configurations of objects that the robot is to manipulate or deal with for purposes of navigation is key to a useful robot system. We recognize three levels at which geometric model information can be put into the robot. The first is user programmed. The second and third are supervised and unsupervised learning, respectively. The latter two are the most interesting, and in the long run, the ones of primary usefulness for any intelligent system. In supervised learning, the object is shown to the robot and identified as to its class association. If it has identifiable sub-parts such as patches of primitive spheres, cylinders, and planes, these are identified by pointing to and labeling each. But the segmentation and representation parameter estimation are carried out by the robot. In unsupervised learning, the robot would figure out for itself what the meaningful sub-parts of an object are. At a more general level, in unsupervised learning a robot would have to figure out for itself just what in a scene are useful or interesting objects. The latter takes considerable processing time, but if a robot has periods of otherwise low demands on it, it can use these periods for unsupervised learning. Situations between the supervised and unsupervised extremes will be common. More generally, as we see it, learning involves determining those models in the defined universe of possible models under consideration that are appropriate to the given information and learning data. The role of the supervisor is solely to provide information that narrows the search space so as to reduce the search and learning times and ambiguity in the end result. This work is being done in projects jointly with Dr. Ruud Bolle of the IBM T. J. Watson Research Center, Yorktown Heights, N. Y., and with Professor Jeffrey Vitter at Brown University.

4 References

- G. Taubin and D. B. Cooper, *Recognition and Positioning of Piecewise Algebraic Objects*, This Proceedings, 1990.
- D. B. Cooper, Y. P. Hung, J. Subrahmonia, *General Model-Based 3D Surface Estimation, Recognition, and Segmentation from Multiple Images*, This Proceedings, 1990.

W. A. Wolovich *Force/Torque Sensing for Object Recognition*, This Proceedings, 1990.

T. L. Dean, T. Camus, J. Kirman, *Sequential Decision Making for Active Vision*, This Proceedings, 1990.

R. M. Bolle and D. B. Cooper, *On Optimally Combining Pieces of Information, with Application to Estimating 3-D Complex-Object Position from Range Data*, IEEE Trans. on PAMI, September, 1986.

Y. P. Hung, D. B. Cooper, B. Cernuschi-Frias, *Asymptotic Bayesian Surface Estimation Using an Image Sequence*, Technical Report LEMS-73, June, 1990.

IU AT UI: AN OVERVIEW OF RESEARCH DURING 1988-90

Narendra Ahuja

Beckman Institute and Coordinated Science Laboratory

University of Illinois

405 North Mathews Avenue

Urbana, Illinois 61801

Abstract

This paper presents an overview of the research in image understanding (IU) at the University of Illinois (UI) conducted since our last report in the proceedings of the 1988 DARPA IU Workshop. During this period (1988-90), we have made progress in four areas: integration in three-dimensional vision, motion analysis, navigation, and parallel algorithms and architectures. Work in each of these areas is reviewed. Research in a more recent area of activity, object recognition, is also summarized.

1 Introduction

A major part of our recent research is in four areas of image understanding. The first area (Sec. 2) deals with integration of multiple image cues in performing image interpretation. These cues capture different aspects of the scene structure, and their integrated analysis leads to a more robust inference about the scene characteristics than possible from individual cues. The second area (Sec. 3) is concerned with our work on interpretation of image sequences showing dynamic scenes. Here we consider the problem of estimating the three-dimensional (3-D) motion parameters and the 3-D surface structure from feature correspondences over a sequence of images and examine the nonrigid motion problem. Projects in the third area (Sec. 4) report work on different components of an evolving 3-D representation and navigation system with the goal of autonomously acquiring, maintaining and using 3-D information about the environment. Finally, in the fourth area (Sec. 5), we summarize our work on a specific multiprocessor architecture that we have proposed for image understanding computations, and on parallel algorithms for a variety of

vision tasks. Section 6 concerns work on object recognition which is a new area of activity at UI. Representative projects in each of these areas are summarized in the following sections. To keep the paper brief, we have minimized discussion of and references to relevant work done by others; such discussion and references are available in our publications cited.

2 Integration

Our goal in this area is to perform 3-D or other interpretation of images, such that the interpretation simultaneously satisfies a range of constraints imposed by the image structure and the model of the scene. To do this, we use different computational processes each of which carries complementary or redundant information derived from different image cues. Image interpretation is the result of a cooperative computation that resolves conflicts and ambiguities arising from the individual processes. We present below three examples of our integration approach; for others, see [2, 7, 8, 22].

2.1 Integrated Passive Stereo

The traditional formulation of the problem of estimating three-dimensional surfaces from stereo images consists of three steps: feature detection, feature matching, and surface interpolation. We have argued in our previous work that the latter two tasks would be more accurately executed in an integrated manner rather than sequentially since they are strongly interdependent. We have reported an algorithm that performs such integration [22]. We have further developed this integration algorithm; it now performs surface fitting directly on depth data (points), rather than on disparity values [16]. Clusters are detected in the three-dimensional distribution of points which result from the matching of feature points in the stereo images. Different clusters correspond to different surfaces. The depth estimation can be performed using arbitrary vergence angles for the cameras.

The integration results in a number of advantages not possible to obtain otherwise. We have explored a new approach which performs a broader integration of stereo; it integrates all three steps instead of just fea-

Parts of this research were supported by grants from the National Science Foundation grants IRI-89-11942 and IRI-89-08225, Air Force Office of Scientific Research under grant AFOSR-90-0061, Army Research Office under grant DAAL 03-87-K-0006, Joint Services Electronics Program under grant N00014-90-J-1270, State of Illinois Department of Commerce and Community Affairs under grant 90-103, Rockwell International, and Eastman Kodak Company.

ture matching and surface interpolation. This is accomplished using an analog formulation in which all three parts of the computation are performed by a monolithic computational structure of dynamical systems. The integration is carried out in parallel by analog signals. This new approach is also expected to be useful for several other low level integration tasks. The details of this work and the advantages of the dynamical systems approach are presented in a separate paper in these proceedings [6].

2.2 Integrated Active Stereo

In this work, we are concerned with the problem of surface reconstruction from stereo images for large scenes having large depth ranges, where it is necessary to aim cameras in different directions, to fixate at different objects, and to construct the global surface map of the scene from small patches. Since the beginning stage of the work reported in [3], we have now developed an active stereo system with a broad range of capabilities. We have carried out the work in two stages. In the first stage, we consider the problem of surface reconstruction of a single object, although the object surface may have large breadth and depth. Consequently, the scan of the object surface can proceed smoothly with no depth discontinuities. We have developed a formalism for the integration of three depth cues: focus, vergence, and stereo disparity. Individually, these sources of depth have their strengths and weaknesses which make them suitable for specific contexts. We have noted that these strengths and weaknesses are quite complementary, and our approach attempts to dynamically combine the use of these cues in a context sensitive manner, so as to take advantage of their strengths while eliminating their weaknesses. There are two components to the approach: selection of a point of fixation on the object, and generating a local surface map in the vicinity of the fixation point. A point is chosen for fixation if it minimizes a certain objective function. The construction of the surface patch in the vicinity of the fixation point uses another objective function which contains as one component the camera calibration parameters; thus, the surface reconstruction can be performed using an unreliably calibrated camera system. This is an important property, since the imaging parameters of a dynamic camera system tend to be different from the true parameters, and the relationship between the two often changes due to mechanical and other errors. By including the camera parameters in the objective function, camera calibration is also carried out dynamically along with surface reconstruction. The overall approach interleaves the processes of image acquisition and surface estimation. The details of this work can be found in [1].

When the scene contains small objects and depth boundaries, the selection of new fixation points becomes difficult if the other objects in the scene are not in focus, and the surface reconstruction of the object cur-

rently in focus has been completed [17]. The location of the object boundary can be obtained from the variation of image sharpness with focus setting. To select a new fixation point, information about the layout of the scene away from the object under fixation is necessary. This presents a dilemma since to acquire such surface structure is the objective of fixation in the first place! To this end, we present an approach to acquiring coarse structural information about the scene in the vicinity of the next fixation point during the current fixation, and utilizing this information for fixation and accurate surface reconstruction in the vicinity of the next fixation point. This work is described in a separate paper in these proceedings [18].

During our work on the focussing process, we have observed that image blurring effects in the vicinity of occlusion boundaries are quite complex. We have shown that blurring processes operating in the vicinity of large depth discontinuities can give rise to spurious and pronounced image details which cannot be explained by previously available blurring models, which usually predict suppression and not creation of image details due to blurring. We have argued that blurring in high-relief scenes should be viewed as a multicomponent process. To this end, we have developed a model of blurring. Extensive experiments with images of real scenes obtained with a CCD camera point toward the qualitative validity of our new blurring model. The details of this work are presented in a separate paper in these proceedings [37].

2.3 Integrating Region, Border and Component Gestalt for Extracting Perceptual Structure

This research concerns perceptual grouping, or goal independent detection of perceptual organization in images. The image tokens that may be grouped include blobs, edge segments, and geometrical features of image regions. One way of understanding grouping phenomena is to eliminate all but one property at a time and examine the effects of that property on grouping. Since dots are without size, orientation, color and shape, dot patterns provide a means for studying the effect of token positions on their grouping, while minimizing the role of nonpositional properties.

The single variable that determines such low level grouping of dots is the relative locations of dots. We have reported in the past on our approach to perceptual grouping of dots that integrates multiple constraints, active at different perceptual levels and having different scopes in the dot pattern. We have extended this integration approach for perceptual grouping to extract perceptual structure in gray level images. The extended approach infers the structure by integrating evidence from region boundaries and region interiors. Although the Gestalt constraints used in our approach are justified on perceptual grounds, we have carried out a quantitative analysis of their significance in defining a per-

ceptual segmentation. We have conducted experiments with a set of dot patterns designed to satisfy to different degrees the different constraints: interior homogeneity, border smoothness, and component compactness. The segmentation results obtained by our algorithm for various combinations of these properties are compared with and are usually the same as perceptual segmentation. Further, we have compared the results of our approach with those obtained by traditional clustering algorithms. These results show that the global optimization of some simple function of interdot distances which is used as the criterion function by the clustering algorithms does not lead to perceptually acceptable segmentation. The clustering algorithms fragment perceptual clusters, merge them, or do both. The results are qualitatively unacceptable, and the types and extents of the errors point to a basic deficiency in the approach rather than to minor problems of adaptation to the data. Our integration approach gives satisfactory performance in all the tests. Details of this work are reported in [4].

3 Motion Analysis

The long-range goal of our research in this area is the understanding of dynamic scenes. The framework we have used consists of three stages: finding feature correspondences in a sequence of frames, determining rigid motion parameters and surface structure from the correspondences, and analyzing and visualizing nonrigid motion.

3.1 Detecting Feature Correspondences

Detecting feature correspondences is difficult due to a wide variety of three-dimensional structural discontinuities and occlusions that occur in real world scenes. We have developed a computational approach to image matching that uses multiple attributes associated with a pixel to yield a generally overdetermined system of constraints, taking into account possible structural discontinuities and occlusions [38, 43]. In our algorithm, intensity, edgeness, and corneriness attributes are used in conjunction with the constraints arising from intraregional smoothness, field continuity and discontinuity, and occlusions to compute dense displacement fields and occlusion maps at pixel grids. A multiresolution multigrid structure, using both bottom-up and top-down flow, is employed to deal with large disparities. Coarser level attributes are obtained by blurring the finer level attributes. The algorithm is tested on real world scenes containing depth discontinuities and occlusions. For short range motion, we obtain time trajectories of feature points, which are defined by sequences of point correspondences across frames.

3.2 Rigid Motion and Structure from Correspondences

Our work in this area can be divided into two categories according to whether the images show long range mo-

tion or short range motion.

In the first category, we have used point correspondences to estimate long range motion and structure from perspective views [44]. For a planar surface, we have developed an algorithm that gives a closed-form solution for motion and structure parameters along with associated errors, for a sequence of monocular perspective images of feature points [39]. The algorithm is simpler and more reliable in the presence of noise than the existing ones. For general surfaces, we have developed an approach that consists of two steps [40]. The first step is estimating the motion parameters using a robust linear algorithm that gives a closed-form solution for motion parameters and scene structure. The second step is improving the results from the linear algorithm using maximum likelihood estimation. Algorithms have been developed using point correspondences as well as line correspondences, and tested on images of real scenes from automatically computed displacement fields.

We have developed an approach to optimal estimation of motion and structure which is applicable whether the type of noise distribution is known or not [41]. For noise distributions with Gaussian model and the uncertainty polyhedron model, we have investigated maximum likelihood estimation. For Gaussian noise, this amounts to minimizing a so called image plane error. For the cases where the type of noise distribution is unknown, it is shown that minimizing image plane error corresponds to an unbiased, minimum variance estimator for a locally linearized system, independent of the type of noise distribution. The performance of the algorithm is compared with a theoretical lower bound called the Cramér-Rao bound. Simulations show that the actual errors are very close to the bound in the case of Gaussian noise. It is shown that in general a batch technique will perform better than a sequential technique for any nonlinear problem. Recursive batch processing is proposed for nonlinear problems that require recursive estimation.

Lines, when available in the images, can serve as more robust features than points. We have obtained a closed-form solution to motion and structure from line correspondences in monocular perspective image sequences [42]. The algorithm requires a minimum of 13 lines over three perspective views. Necessary and sufficient conditions for degenerate spatial line configurations have been derived. Simulations are performed which show the performance of the algorithm in the presence of noise. An approach to optimal estimation of motion and structure using line correspondences is presented in [45]. Simulation results show that, contrary to the general view, a line-based algorithm is not necessarily more unstable than a point-based algorithm.

[23] describes methods of formulating motion problems as the solution of simultaneous polynomial equations, and discusses the use of homotopy methods for solving the polynomial systems. In [31], an example of using a priori information to narrow down the search

space in a 3D object recognition problem is presented. The main goal of this work is to test how well existing feature extraction/matching and motion estimation algorithms work on outdoor scenes.

In the second category, we analyze an image sequence showing short range motion. We have developed an algorithm for estimating motion and structure from orthographic projections [19]. The objects can have arbitrary shapes. The algorithm uses closed form expressions to find estimates of the motion and the structure parameters, and can accept any number of points over any number of frames as input. The algorithm consists of two steps: (i) finding the trajectories of feature points over the image sequence, and (ii) grouping these trajectories into sets corresponding to distinct rigid objects, and finding the structure and motion of each of these rigid objects. We have shown that the accuracy of the algorithm improves with larger total rotations, with more intermediate frames, with smaller amounts of input noise, and to a limited extent with the number of points.

3.3 Nonrigid Motion

Many scenes of interest contain nonrigid objects, e.g., people. We have reported on an algorithm for analyzing the motion of objects consisting of jointed rigid parts (e.g., robot arms) [21]. More recently, we have considered scenes in which objects undergo nonrigid motion. Some of our initial thoughts on nonrigid motion analysis are contained in [24] and [25].

Many of the techniques and concepts for rigid motion analysis and object recognition are also useful for nonrigid objects. The few projects on nonrigid motion reported in the literature have been concept and technique oriented. To complement these efforts, we have taken an application-motivated approach. Three important problems in elastic and fluid motion are picked: modeling and analysis of heart wall motion, human face/head motion analysis and synthesis in model-based image compression, and studying the evolution of coherent structures in fluid motion. The starting point of our research is realistic data from these problems.

In our work on heart motion, we have concentrated on the left ventricle (LV). We have worked mainly with biplane cineangiograms. 20–40 vessel bifurcation points are identified and tracked over the successive image frames. By stereo triangulation, the 3D coordinates of these points are determined at each time instant. We have devised algorithms to determine the global motion of the LV, and after compensation for global motion to determine the local motion in terms of stress tensors. We have also experimented with several methods of visualization. Some of our results are described in [20, 9, 10].

The theme of our research on human face/head motion is modeling, analysis, and synthesis, and it is motivated by applications to visual communication (model-based image compression) and time-varying pattern

recognition (e.g., computer lip reading). The key issue in both analysis and synthesis is the static and dynamic modeling of the human face/head. The head can be reasonably modeled by a rigid ellipsoid. The face, being nonrigid, is much more complicated; it can be approximated by triangular patches where the vertices of the triangles can have restricted motions. To be realistic, perhaps around 400 triangular patches are needed. Our effort has been in setting up the mechanism for visualizing human face/head motion. A crude 3D model of a generic face (containing 80 triangles) has been constructed and stored. We have developed an interactive procedure for fitting the model to a particular person using one or more images of the person's face—this may be called the customization of the face model. We can move the model globally (e.g., to make the head nod) and locally (e.g., to make the face smile) and generate 2D sequences for visualization.

In our work on fluid motion, we are interested in developing automated analysis and visualization techniques for turbulent fluid flow with a view toward applications in the acquisition of flow data, the interpretation of complicated fluid motions, and the active control of flows and flow systems. Our main emphasis is on the detection and recognition of vortices and the tracking of their evolution. We have worked on vortex modeling. Using a linear prediction approach, we have developed a mathematical model for vortices. By changing the parameters in the model, we can generate vortices with various degrees of deformation. Some of the results are contained in [46].

4 3-D Occupancy Maps from Images and Navigation

The goal of our work in 3-D mapping is to develop algorithms for acquiring 3-D information about the occupancy of space by objects. We are concerned with only a coarse 3-D representation of filled/empty space, not with representing fine shape details of occupied space. There are three different aspects to representation of spatial occupancy: initial *generation* of the representation for a given scene, its *maintenance* as the objects in the scene move, and its *use* in environment manipulation including tasks such as planning trajectories of moving objects through the scene. Our work has addressed all three of these aspects. Our emphasis recently has been on the generation and navigation parts. We have reported on algorithms for generating the octree representation of an object from its silhouettes seen in orthographic views [5] and perspective views [36].

In navigation, we have concentrated on the problem of path planning, i.e., deriving an efficient and collision free trajectory to move an object from a given source location/orientation to a given destination location/orientation through an environment with given occupancy map. To represent the free space, we have used a potential field. The problem is divided into two stages. First, a candidate topological path is selected. Second,

the candidate path is cost-optimized to derive the final path and orientations of the moving object. The details of our path planning work can be found in [28]. A survey of the state of the art in autonomous path planning among obstacles is presented in [29]. Recently, we have begun to use the more realistic, Newtonian potential function which improves the computational efficiency [15]. We have used the new potential field to represent 2-D objects and obstacles consisting of line segments. The computational efficiency arises from the fact that we have closed-form expressions for the potential field as well as some other gradient related quantities. The availability of the closed form expressions eliminates the need for search based evaluation of the risk of collision which requires discretization of the object and obstacles. Further, it becomes possible to use binary search to find the optimal object configurations for path planning.

5 Parallel Architectures and Algorithms

To design an efficient computer architecture requires taking into account the characteristics of the algorithms to be executed by the architecture. A major feature of image data is its planar nature, and an important feature of many image computations is that they are spatially local. Together these two characteristics imply that parallel subtasks may be generated by using the divide-and-conquer paradigm to perform computations on subimages, in parallel, and by merging their results. The degree of exploitable parallelism in computer vision tasks varies with time and image position. Therefore, an architecture for vision must be highly flexible and modular. We have reported [3, 14] on a highly flexible multiprocessor system called NETRA for image computations which we have been developing. NETRA is highly reconfigurable and does not involve the use of complex interconnection schemes. The topology of this multiprocessor is recursively defined, and hence, is easily scalable from small to large systems. It has a tree-type hierarchical architecture each of whose leaf nodes consists of a cluster of small but powerful processors connected via programmable crossbar with selective broadcast capability.

We have simulated NETRA on a hypercube multiprocessor and evaluated the performance of one processor cluster for stereo computations [12]. The stereo algorithm selected for implementation requires computation of two-dimensional Fast Fourier Transform, template matching, histogram computation and least-squares surface fitting. The superior performance of NETRA has been demonstrated by comparing the results with those from a similar implementation on the hypercube. We have developed techniques for static partitioning of data for the data independent tasks such as two-dimensional Fast Fourier Transform, and dynamic scheduling and load balancing for the data dependent tasks of feature matching and disambiguation. As an indicator of the significance of dynamic load bal-

ancing over the static one, the improvement in the performance using dynamic load balancing is by more than a factor of two for the stereo algorithm when implemented on a hypercube multiprocessor [11]. We have also evaluated our load balancing techniques using a motion estimation algorithm implemented on a hypercube multiprocessor [13]. The motion estimation task consists of the following steps: 1) extraction of features, 2) stereo matching, 3) matching between images obtained at different times, and 4) computation of motion parameters. The performance gain using these techniques is again shown to be significant while the overhead in using them is minimal.

6 Object Recognition

This section summarizes our work on object recognition.

6.1 3D Curved Objects

Our goal here is to develop algorithms capable of recognizing and locating in a single image instances of curved 3D objects modelled by parametric patches and their intersection curves. We focus on the following three areas:

Object representation: We have developed and implemented a new algorithm for computing the exact aspect graph of solids of revolution [30]. We have extended this work to objects modelled by parametric patches and their intersection curves, and have demonstrated a preliminary implementation [34].

Pose determination: We have already demonstrated recognition and positioning of solids of revolution from their monocular image contours using elimination theory [33]. We are currently investigating new methods for positioning curved 3D objects from contour orientation and curvature and from image intensity and gradient [35].

Matching strategies: For curved objects, image features such as contours or t-junctions are not the projections of particular object features; this makes purely object-centered feature matching difficult, maybe impossible. We are investigating the qualitative and quantitative constraints imposed by image features to match these features to aspect graphs.

6.2 Evidential Reasoning

We have implemented a recognition system based on the Dempster-Shafer formalism [27]. The system constructs belief functions that define credibility in candidate hypotheses as a function of the differences between model and scene data. We have implemented belief functions based on the quantitative difference in feature attributes, relational consistency, and aspect consistency. In each case, the belief functions are normalized confidence values, these values being obtained by specifying a nonlinear mapping from quantitative differences between model and scene data to confidence values.

A major criticism the Dempster-Shafer formalism is that the complexity of Dempster's rule is, exponential in

the number of hypotheses. We have shown [26] that for our approach, the belief functions constructed belong to a distinguished family known as disjoint belief functions, and that, in this case, Dempster's rule has a polynomial time implementation.

A further application of our recognition system is planning sensing strategies to reduce ambiguity in scene interpretation. We have developed a system [26] that proposes and evaluates sensing operations based on maximum reduction of ambiguity. The worst case ambiguity for a proposed sensing operation is computed by predicting features that would be observed for each of the currently held hypotheses. These predicted features are used to derive resulting scene interpretations. The worst case ambiguity over all currently held hypotheses is a measure of the effectiveness of the proposed sensing operation.

References

- [1] L. Abbott and N. Ahuja, Surface Reconstruction by Dynamic Integration of Focus, Camera Vergence and Stereo, *Second International Conference on Computer Vision*, Tarpon Springs, December 5-8, 1988, 532-543.
- [2] N. Ahuja, Texture Analysis, S. Shapiro (ed.), *Encyclopedia of Artificial Intelligence*, Wiley, 1990.
- [3] N. Ahuja and T. S. Huang, IU at UI: An Overview and an Example on Shape from Texture, *Proc. DARPA Image Understanding Workshop*, Cambridge, April 6-8, 1988, 222-253.
- [4] N. Ahuja and M. Tuceryan, Extraction of Basic Perceptual Structure in Dot Patterns: Integrating Region, Boundary and Component Gestalt, *Computer Vision, Graphics and Image Processing*, December 1989, 304-356.
- [5] N. Ahuja and J. Veenstra, Generating Octrees from Object Silhouettes in Orthographic Views, *IEEE Trans. Pattern Analysis and Machine Intelligence*, February 1989, 137-149.
- [6] E. Altman and N. Ahuja, A Dynamical Systems Approach to Integration in Stereo, *these proceedings*.
- [7] D. Blostein and N. Ahuja, A Multiscale Region Detector, *Computer Vision, Graphics and Image Processing*, January 1989, 22-41.
- [8] D. Blostein and N. Ahuja, Shape from Texture: Integrating Texture Element Extraction and Surface Estimation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, December 1989, 1233-1251.
- [9] C.W. Chen and T.S. Huang, Epicardial motion and deformation estimation from coronary artery bifurcation points, Tech. Note ISP-1020, Coordinated Science Laboratory, Univ. of Illinois at Urbana-Champaign, April 25, 1990. A revised version to appear in *Int. Jour. of Imaging Systems and Technology*, Dec. 1990.
- [10] C.W. Chen, T.S. Huang, and M. Arrott, Analysis and visualization of heart motion, *Proc. SPIE/SPSE Symp. on Electronic Imaging Science and Technology*, Conf. on Biomedical Imaging Processing, II, Feb. 24-March 1, 1991, San Jose, CA.
- [11] A. N. Choudhary, S. Das, N. Ahuja and J. Patel, Surface Reconstruction from Stereo Images: An Implementation on a Hypercube Multiprocessor, *Fourth Conference on Hypercube Concurrent Computers and Applications*, March 1989.
- [12] A. N. Choudhary, S. Das, N. Ahuja, and J. H. Patel, A Reconfigurable and Hierarchical Parallel Processing Architecture: Performance Results for Stereo Vision, *1990 International Conference on Pattern Recognition - Computer Vision*, 389-393.
- [13] A.N. Choudhary, M.K. Leung, T.S. Huang, and J. Patel, Parallel implementation and evaluation of motion estimation algorithms on a distributed memory multiprocessor using knowledge based mappings, *Proc. 10th ICPR*, June 16-21, 1990, Atlantic City, NJ, 337-342.
- [14] A. N. Choudhary, J. H. Patel and N. Ahuja, Performance Evaluation of NETRA, in *Parallel Architectures and Algorithms for Image Understanding*, (Ed. V. K. Prasanna Kumar), Academic Press, to appear.
- [15] R. Chuang and N. Ahuja, Path Planning Using the Newtonian Potential, *IEEE Trans. on Robotics and Automation*, submitted.
- [16] S. Das, A. L. Abbott and N. Ahuja, Surface Reconstruction from Focus and Stereo, *Proc. 5th International Conference on Image Analysis and Processing*, Positano, Italy, September 1989.
- [17] S. Das and N. Ahuja, Integrating Multiresolution Image Acquisition and Coarse-to-fine Surface Reconstruction from Stereo, *IEEE Workshop on Interpretation of 3D Scenes*, Austin, November 26-29, 1989, 9-15.
- [18] S. Das and N. Ahuja, Integrated Multiresolution Image Acquisition and Surface Reconstruction from Active Stereo, *these proceedings*.
- [19] C. Debrunner and N. Ahuja, A Hankel Matrix Based Motion Estimation Algorithm, *1990 International Conference on Pattern Recognition - Computer Vision*, 384-389.
- [20] B. Goldgof, H. Lee, and T.S. Huang, Parameter estimation of the heart motion from angiography data, *Proc. SPIE Conf. on Biomedical Image Processing*, vol. 1245, Feb. 12-13, 1990, Santa Clara, CA.
- [21] D. Goldgof, T. S. Huang and H. Lee, Motion Estimation of Nonrigid Objects, *Proc. IEEE Conf.*

- Computer Vision and Pattern Recognition*, June, 1988, Ann Arbor.
- [22] W. Hoff and N. Ahuja, Surfaces from Stereo: Integrating Feature Matching, Disparity Estimation and Contour Detection, *IEEE Trans. Pattern Analysis and Machine Intelligence*, February 1989, 121-136.
 - [23] R.J. Holt, A.N. Netravali, and T.S. Huang, Experience in using homotopy methods to solve motion estimation problems, *Proc. SPIE*, vol. 1251, Curves and Surfaces in Computer Vision and Graphics, Feb. 13-15, 1990, Santa Clara, CA.
 - [24] T.S. Huang, Modeling, analysis, and visualization of nonrigid objects, *Proc. 10th ICPR*, June 16-21, 1990, Atlantic City, NJ.
 - [25] T.S. Huang, Image processing in the broad sense (Invited), *Proc. Bilkent Int. Conf. on New Trends in Communication, Control, and Signal Processing*, July 2-5, 1990, Ankara, Turkey.
 - [26] S. A. Hutchinson and A. C. Kak, Planning Sensing Strategies in a Robot Work Cell with Multi-Sensor Capabilities, *IEEE Trans. on Robotics and Automation*, 5(6):765-783, Dec. 1989.
 - [27] S. A. Hutchinson, R. L. Cromwell and A. C. Kak, Applying Uncertainty Reasoning to Model Based Object Recognition, *Proc. of the IEEE Conf. on CVPR*, 1989.
 - [28] Y. Hwang and N. Ahuja, Path Planning using a Potential Field Representation, *IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, June 4-8, 1989, 569-575.
 - [29] Y. Hwang and N. Ahuja, Gross Motion Planning - A Survey, *ACM Computing Surveys*, submitted.
 - [30] D.J. Kriegman and J. Ponce, Computing exact aspect graphs of curved objects: solids of revolution, In *Proc. IEEE workshop on interpretation of 3D scenes*, Austin, TX, Nov. 1989.
 - [31] M.K. Leung and T.S. Huang, Detecting wheels of vehicles in stereo images, *Proc. 10th ICPR*, June 16-21, 1990, Atlantic City, NJ.
 - [32] M. K. Leung, Y. C. Liu and T. S. Huang, Experimental Results of 3D Motion Estimation Using Images of Outdoor Scenes, *these proceedings*.
 - [33] J. Ponce and D.J. Kriegman, On recognizing and positioning curved 3D objects from image contours, In *Proc. Image Understanding Workshop*, pages 461-70, may 1989.
 - [34] J. Ponce and D.J. Kriegman, Computing exact aspect graphs of curved objects: parametric patches, In *Proc. Am. Assoc. Art. Intell.*, July 1990.
 - [35] J. Ponce and D.J. Kriegman, Elimination theory and computer vision, In *Workshop on the integration of numerical and symbolic computing methods*, Saratoga Springs, NY, July 1990.
 - [36] S. Srivastava and N. Ahuja, Octree Generation from Object Silhouettes in Perspective Views, *Computer Vision, Graphics and Image Processing*, January 1990, 68-84.
 - [37] C. Thang and T. S. Huang, Image Blurring Effects due to Depth Discontinuities, *these proceedings*.
 - [38] J. Weng, N. Ahuja, and T. S. Huang, Two-view matching, in *Proc. Second International Conference on Computer Vision*, Florida, Dec. 1988, pp. 64-73.
 - [39] J. Weng, N. Ahuja, and T. S. Huang, Motion and structure from point correspondences: a robust algorithm for planar case with error estimation, in *Proc. International Conference on Pattern Recognition*, Rome, Italy, 1988, pp. 247-251.
 - [40] J. Weng, N. Ahuja, and T. S. Huang, Closed-form solution + maximum likelihood: a robust approach to motion and structure estimation, in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Ann Arbor, Michigan, June 5-9, 1988, pp. 381-386.
 - [41] J. Weng, N. Ahuja, and T. S. Huang, Optimal motion and structure estimation, in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June, 1989, pp. 144-152.
 - [42] J. Weng, Y. C. Liu, T. S. Huang, and N. Ahuja, Estimating motion/structure from line correspondences: a robust linear algorithm and uniqueness theorems, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Ann Arbor, Michigan, June 5-9, 1988, pp. 387-392.
 - [43] J. Weng, T. S. Huang and N. Ahuja, Motion from images: image matching, parameter estimation and intrinsic stability in *Proc. IEEE Workshop on Visual Motion*, Irvine, CA, March 20-22, 1989, pp. 359-366.
 - [44] J. Weng, T. Huang and N. Ahuja, Motion and Structure from Two Perspective Views: Algorithm, Error Analysis, and Error Estimation *IEEE Trans. Pattern Analysis and Machine Intelligence*, May 1989, 451-476.
 - [45] J. Weng, T. S. Huang and N. Ahuja, Motion and structure estimation from line matches: performance obtained and beyond, in *Proc. Tenth International Conference on Pattern Recognition*, Atlantic City, New Jersey, June 17-21, 1990, 168-172.
 - [46] J. Zhong and T.S. Huang, Detection and recognition of vortices in fluid flow, *Proc. Amer. Phys. Soc., Div. of Fluid Mechanics Meeting*, Nov. 16-18, 1990, Ithaca, NY.

Vision-Based Navigation

William B. Thompson,
Computer Science Department
University of Minnesota
Minneapolis, MN 55455

Abstract

Formidable challenges are faced by developers of autonomous systems for vision based navigation in general outdoor environments. Basic research efforts at the University of Minnesota are focussed on the localization problem, a key task of vision based navigation. The localization task is concerned with the visual determination of viewpoint and viewing direction relative to a map. Our current efforts are concerned with the process of establishing the correspondence between a set of map features and image features as one aspect of the localization task.

1 Introduction

We are initiating an interdisciplinary investigation of the interactions between visual perception and navigation. Navigation problems typically involve purposeful maneuvering through unfamiliar terrain in order to accomplish some goal. Our emphasis is principally on navigation through *large-scale space* – a space whose structure is of a significantly larger scale than that of the observer. (This definition is closely related to that of [Kuipers and Levitt, 1988].) There are two important perceptual consequences of navigation through large-scale space. The first is the difficulty of estimating accurate locations of objects and orientations of surfaces at a significant distance from the observer. The second is that only a small portion of the space is typically visible at any one time. As a result, planning activities must apply some form of cartographic information that describes relevant aspects of the terrain.

Fundamental problems in navigation involve determining an agent's present position and orientation on a map, determining the desired route, and performing position updates with respect to the map as the route is traversed. While non-visual navigation aids such as inertial guidance and satellite-based locating systems can provide relatively precise information on position and/or orientation, they have characteristics (e.g., drift or susceptibility to countermeasures) that make them inappropriate for robotic systems where the mission requires high precision and robustness under adverse conditions. Active range sensing also has important limitations. For example, sonar is appropriate only for relatively unclut-

tered indoor environments. Laser sensing, while offering utility in some outdoor navigation problems, loses effectiveness as distances increase. In some situations, active sensing is simply not possible. Our work assumes that passively sensed visual imagery is the primary input for map matching and positional updating.

Research on vision-based navigation in outdoor environments has primarily focused on lower-level problems such as road following (e.g., [Thorpe *et al.*, 1988]). Work on using maps has concentrated on closed-loop feedback methods in which map data, knowledge of current position, and dead-reckoning are used to predict visual features occurring after a short time interval (e.g., [Fennema *et al.*, 1988, Ernst and Flinchbaugh, 1989]). Little research within the image understanding community has addressed navigation problems that arise due to the increased uncertainty associated with locomotion over longer time intervals, greater distances, and unfamiliar terrain. The matching of maps and aerial imagery has been extensively studied (e.g., [McKeown and Denlinger, 1984] and many others), but few of the low-level image understanding algorithms currently available are directly applicable to the outdoor, horizontally viewed imagery necessary to support mobile robot navigation.

More so than most other problems in vision, navigation seems to require substantial problem solving closely integrated with lower-level perceptual analysis. As a result, our research is combining a computational analysis of navigation with a study of the methods used by people experienced at solving difficult navigation problems (see [Heinrichs *et al.*, 1989]). The computational analysis provides the power of formal tools and points towards implementable computer algorithms. Experience with expert navigators provides insight into efficient problem solving strategies for image understanding that would be difficult to discover in any other manner.

2 Localization

Localization is the process of establishing a match between particular locations in the environment and the corresponding locations on a map. Locations of interest can involve the viewpoint (current position) and/or distant features. The methods necessary to solve localization problems vary significantly depending on how much is known about the actual viewing position. Computational analysis has in the past been limited almost



Figure 1: Image.

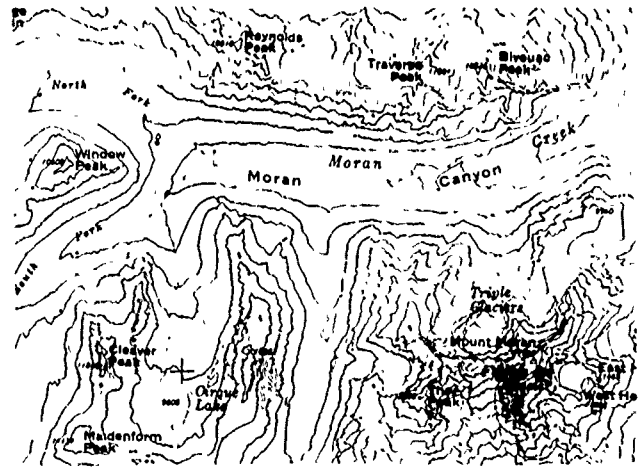


Figure 2: Topographic map.

exclusively to *updating* problems, where the task is to maintain a sense of the current position with respect to a map as the current position changes due to locomotion. Updating problems are typically solved by developing expectations about the imagery likely to be available at a future point in time, comparing these expectations with the actual sensed data, and finally revising the position estimate to account for any discrepancies.

Updating is not sufficient, however, to solve all localization problems when a mobile agent travels through large-scale, outdoor spaces. Problems arise due to a frequent lack of distinctive landmarks, occlusion and disocclusion of features, the weak correspondence between low-level image features and topography, terrain in which small changes in viewpoint produce substantial changes in the imagery, and difficulties revising estimates of current position when expected and actual views differ by more than a small amount. In addition, updating requires continuous image analysis over a wide field of view and this is often not possible.

Reliable localization requires the ability to solve *drop-off* problems involving substantial initial uncertainty in viewing location and/or direction.¹ Figures 1 and 2 show an example of a map and image that might be involved in the solution to such a problem. Localization can be used to identify the prominent canyon visible in the image, estimate viewpoint position, and determine the map locations of peaks and ridges visible in the image. Drop-off problems, particularly in outdoor scenes, are an aspect of navigation that has received almost no study by the robotics and image understanding communities, save in situations in which unique, identifiable landmarks are available and visually recognizable.

¹The name comes from the extreme case in which an observer is "dropped off" into a totally unfamiliar environment, though drop-off methods are needed for a far broader class of problems.

3 A Formalism for Drop-Off Problems

We have created a formal conceptual structure within which drop-off problems can be specified and analyzed [Thompson *et al.*, 1990]. This formalism has much in common with several successful approaches to object recognition. In the case of localization, the task is to recognize a "model" that describes the topographic features visible from a particular vantage point. As with object recognition, the solution involves the selection of appropriate subsets of image features to match against models, the selection of appropriate models, and establishment of correspondences between model and image features using alignment techniques [Grimson, 1990]. Unlike object recognition, however, a discrete, predefined set of models is not available, since even modest uncertainty in location leads to a very large number of possible, distinct viewpoints. As a result, drop-off problems require that models be assembled from map data on a problem specific basis before they are matched to images. Localization problems also involve a unique form of alignment. Topography creates an approximately horizontal plane that is viewed on end. Much of the three-dimensional structure is lost due to the horizontal projection involved in imaging. Only certain sorts of features can be simultaneously located in horizontally viewed imagery and the "downward looking" representation of terrain contained in a map.

This conceptual structure accounts nicely for several strategies that have been observed in expert map users. For example, we know that on the ground, proficient map users most often start solving localization problems with a visual examination of the environment and then proceed to an examination of the map. Fighter pilots, on the other hand, start with the map and only after selecting appropriate landmarks look out to search the environment. Our analysis shows why these two strategies are appropriate in the two different situations. The formalism also indicates how certain difficulties in visually estimating spatial relationships between topographic features can be aided by an appropriate choice of viewpoint – suggesting that there may be important interac-

tions between path planning and localization for a mobile robot.

We are building a computer system within which to explore specific aspects of the localization task. In this system, knowledge bases hold explicit descriptions of possible topographic structures, information about map and image extracted from each individual problem, and a listing of currently active hypotheses about which map features match which image features and what the viewing position and direction actually is. A set of control structures focuses search to manage computational complexity and provides procedures for recognizing features, assembling configurations of features, and posting, evaluating, refining, and accepting or rejecting hypotheses about correspondences and viewpoint. As with a number of approaches that have been proposed for object recognition, we emphasize the importance of establishing potential correspondences between image and map (model) features prior to the estimation of the viewing transformation relating map (model) and image.

Lower-level image understanding aspects of the system draw on techniques for limiting complexity in the matching of map and image features together with an analysis of how knowledge about topography can be used to aid in the segmentation of outdoor scenes. Expert map users employ several heuristics for selecting viewpoint invariant features with which to establish a correspondence between map and image. Some are as simple as looking for a distinctive ordering of primitive features physically adjacent and lying along a straight line. These heuristics appear to be directly applicable to computer implementations. At a more primitive level, information about the distribution of slopes and ridge rise angles can be used to generate constraints on the orientation of occluding contours that will appear in the image. These constraints lead to simple techniques for improving the reliability of edge detectors which form the first step in extracting topographic features from an image. Future work will improve upon these methods and study the problems associated with the visual recognition of features needed to solve localization problems.

4 Related Research Activities

In the coming year, two additional research activities will be pursued. The first deals with low-level issues involving the perception of spatial information in outdoor scenes. The second addresses higher-level questions about the interaction of problem solving and perception for navigation.

Few of the "shape-from-X" techniques that have been so extensively investigated over the last decade are all that effective in outdoor environments, particularly with ground-level imagery. Calculating depth from stereo or motion becomes less feasible as the features of interest become more distant. Shape from shading or texture is confounded by the complex surface coverings that are usually present. A study of the perception of large-scale space will examine limits on the performance of vision systems intended to support outdoor navigation. As part of this work, we are interested in learning more about the visual cues used by people when analyzing large-scale,

outdoor scenes. (Somewhat surprisingly, no references appear in the psychology literature regarding perceptual competence in the sorts of scenes relevant to outdoor, non-urban navigation.) A key problem to be solved is the development of formal representational tools capable of describing the information that is available in images of this sort [Thompson and Kearney, 1986].

The ambiguities and inaccuracies inherent in vision require the use of specialized navigation problem-solving techniques. Understanding the knowledge and strategies involved in the navigation process is essential to understanding how vision relates to the navigation task. Close integration of low-level vision and higher-level reasoning will be needed. In particular, the strengths and weaknesses of computer vision methods applied to the problems in which we are interested should be identified and formalized so that compensatory reasoning strategies can be developed.

References

- [Ernst and Flinchbaugh, 1989] M. D. Ernst and B. E. Flinchbaugh. Image/map correspondence using curve matching. In *AAAI Symposium on Robot Navigation*, pages 15-18, March 1989.
- [Fennema et al., 1988] C. L. Fennema, Jr., E. M. Riseman, and A. R. Hanson. Planning with perceptual milestones to control uncertainty in robot navigation. In *Proceedings of the SPIE*, 1988.
- [Grimson, 1990] W. E. L. Grimson. *Object Recognition by Computer: The Role of Geometric Constraints*. The MIT Press, 1990.
- [Heinrichs et al., 1989] M. R. Heinrichs, D. R. Montello, C. M. Nusslé, and K. Smith. Localization with topographic maps. In *Proceedings of the AAAI Symposium on Robot Navigation*, pages 29-32, March 1989.
- [Kuipers and Levitt, 1988] B. Kuipers and T. S. Levitt. Navigation and mapping in large-scale space. *AI Magazine*, 9(2):25-43, 1988.
- [McKeown and Denlinger, 1984] D. M. McKeown, Jr. and J. L. Denlinger. Map-guided feature extraction from aerial imagery. *IEEE Workshop on Computer Vision: Representation and Control*, pages 205-213, 1984.
- [Thompson and Kearney, 1986] W.B. Thompson and J.K. Kearney. Inexact vision. *Proc. Workshop on Motion: Representation and Analysis*, pages 15-21, 1986.
- [Thompson et al., 1990] W. B. Thompson, H. L. Pick, Jr., B. H. Bennett, M. R. Heinrichs, S. L. Savitt, and K. Smith. Map-based localization: The "drop-off" problem. 1990. These Proceedings.
- [Thorpe et al., 1988] C. Thorpe, M. H. Herbert, T. Kanade, and S. A. Shafer. Vision and navigation for the Carnegie-Mellon Navlab. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10:362-373, 1988.

SECTION II

TECHNICAL PAPERS

NEW RESULTS IN SHAPE FROM SHADING

J. Oliensis

Computer and Information Science Department
University of Massachusetts at Amherst *

Abstract

It is argued that the solutions to shape from shading are often well-determined, with little or no ambiguity, for general illumination direction. For the case of illumination symmetric around the viewing direction, it is proven that there is a *unique* shape solution for *general* images. Also, the long open question whether the image of the occluding boundary significantly constrains the shape solutions is answered in the negative. An image example is presented which demonstrates, for the case of general light source direction, that shape from shading can be well-posed and ill-posed simultaneously: although the shape corresponding to most of the image example is uniquely determined, the shape for a specified small image region is ill-determined. Such 'ill-posed' regions are probably small fractions of images in general, but can occur frequently. The existence of solutions is also studied. It is argued that for almost all images, i.e., for almost all intensity functions, no solution exists. Finally, a new local algorithm for reconstructing shape from shading using a general quadratic surface model is presented.

1 Introduction

Shape from shading has traditionally been assumed to yield multiple object solutions. In fact, it has usually been considered an *ill-posed problem*, with an infinite number of solutions. These assumptions have previously been shown to be false, but only for quite restricted lighting conditions and classes of images. Thus, the uniqueness theorem of Bruss [2], which is the most substantial result of this type, requires the reflectance function to be symmetric around the optical axis, and the image to contain exactly one singular point, that is, a single maximally bright point.

The question whether shape from shading is ill- or well-posed is important, because the traditional approach to reconstructing shape employs *regularization* techniques, implicitly assuming the problem to be ill-posed. If the problem is actually well-posed, then regularization is unnecessary, and should be avoided since it

can lead to a distortion of the recovered shape. It is also important to understand what the constraints on the solutions to shape from shading are, especially if they are significant enough to render the problem well-posed. By incorporating all available constraints in a shape reconstruction algorithm, one can hope to improve the robustness of shape recovery.

This paper summarizes the first arguments showing that the solutions to shape from shading can be well-determined, or even uniquely determined, for *general* images, and for illumination of a Lambertian surface from a general direction. Our main result is a proof that for generic images, and the reflectance functions considered by Bruss, the solution to shape from shading is actually *unique* [7]. This is the first uniqueness result not limited to a restricted class of images. For general illumination direction, it is demonstrated that the solution is not necessarily unique, but that it is frequently well-determined over much of the image, with little ambiguity [8]. These are the first such results valid for general illumination direction.

Also, the constraints determining the solutions are analyzed, and it is shown that singular points provide strong constraints, but that the image of the occluding boundary does *not* [8]. This is surprising since the surface orientations are determined at the image of the occluding boundary. Lastly, we show that shape from shading, over different image regions, can be well-posed and ill-posed simultaneously [8]. An example is presented for which the shape is uniquely determined over most of the image, but ill-determined corresponding to a particular small image region bordering the image boundary. It is argued in [7] that such 'ill-posed' image regions can occur frequently, but that they are probably small fractions of images in general. The implication is that regularization is sometimes necessary for shape reconstruction, but should be used judiciously, that is, only in the appropriate regions. Also, shape should ideally be reconstructed from the interior of the image outwards, in order to avoid propagating errors and instabilities associated with the possible ill-posed regions at the image boundary.

The existence of solutions to shape from shading is also studied. It is argued that for almost all images, i.e., for almost all intensity functions $I(x, y)$, *no solution to shape from shading exists* [7]. Thus, true images of

*This work was supported by the Defense Advanced Research Projects Agency under grants F30602-87-C-0140 and DACA76-89-C-0017, and by the National Science Foundation under grant DCR-8500332.

objects are very special intensity functions. We argue that a true image can be converted into an 'impossible' one, with no object solution, by a small perturbation of its intensities. The only previous non-existence result was limited to a narrow set of images [13].

Although our result, like that of [13], has been derived just for reflectance functions symmetric around the optical axis, it probably holds in very general lighting conditions. The possibility that no consistent solution exists provides an important failure criterion for shape from shading. If the method is applied to an inappropriate image, then this will probably be signaled by the non-existence of any consistent solution, showing that the assumptions made about the scene were incorrect.

Finally, we describe a new local algorithm for recovering shape, which uses a general quadratic surface model as a local approximation of the imaged surface. The model is recovered from the local image data. Previous algorithms used rather special local surface models, for instance the spherical model of Pentland [10], or the restricted quadratic model of Ferrie and Levine [6].

2 The Uniqueness Theorem

In the following sections we present our uniqueness theorem, and sketch its proof.

First, we give some definitions. The following are standard: A *singular point* in the image is a point of maximal brightness for the given reflectance function. The surface orientation corresponding to a singular point is uniquely determined. For a smooth, closed, object, the *occluding boundary* is defined to be the set of all object points at which the surface normal is perpendicular to the optical axis; the *limb* is the image of the occluding boundary. Finally, a smooth, closed object is *non-self-occluding* if all points on the limb are also on the boundary of the image region containing the projected object. This is actually just a slight extension of the standard, common-sense meaning of non-self-occluding. In addition to the usual cases of self-occlusion, it excludes cases where the object 'virtually' self-occludes—where the surface normal becomes perpendicular to the optical axis with no actual self-occlusion.

The following definitions are new. A *Bruss reflectance function* $R(p, q)$, where p, q are the usual partial derivatives of the depth z , is defined to be one which satisfies: 1) R is a smooth, non-negative, function depending only on $p^2 + q^2$ (thus, it is symmetrical about the optical axis \hat{z}), 2) R attains a unique global maximum at $p = q = 0$, with the surface normal to the optical axis, 3) the derivative of R with respect to its argument $p^2 + q^2$ is less than zero at this point, 4) $R \rightarrow 0$ as its argument goes to infinity, so that R vanishes on the occluding boundary, 5) R is monotonic (thus, the image irradiance equation can be inverted, and transformed into eikonal form). A typical Bruss reflectance function corresponds to the illumination of a matte or Lambertian surface from the viewer direction.

A *generic* class is one containing essentially all instances, apart from a few special cases. Moreover, for any special case not contained in this class, an infinitesimal

perturbation will yield an instance contained in the class. Thus, a generic class of images contains essentially all images. The special cases not contained in this class can be avoided by infinitesimal perturbations of the object shape, or by an infinitesimal object rotation. Similarly, a generic property is true of essentially all instances, and false only in special and unstable cases.

In [7] it is shown that the class of *structurally stable* images is a generic class of images. This is generalized to the case of general illumination direction in [8]. A structurally stable image is one satisfying a certain set of properties, for instance it must contain a finite number of singular points. The exact definition of this concept is given later. Also, among images of closed, non-self-occluding objects, the images with *smooth limb* are a generic class [14]. In other words, essentially all images of non-self-occluding objects have smooth limb curves.

Theorem: Assume 1) an image of a closed, smooth, non-self-occluding, genus zero object is produced by orthographic projection, 2) the reflectance function is a Bruss reflectance function, 3) the object is completely contained in the image, and the limb is a smooth, closed, curve (this is generically true from the above), 4) the image belongs to the generic class of structurally stable images. Then the visible surface of the original object is the unique solution to shape from shading corresponding to a closed object.

Alternatively, the conclusion of this theorem can be restated as follows. The pose of a non-self-occluding surface is defined to be *accidental* if an infinitesimal rotation will cause the surface to self-occlude. **Theorem:** Assume conditions 1-4 above. Then the visible surface of the original object is the unique solution to shape from shading whose pose is non-accidental.

This uniqueness result may seem to conflict with the well-known, two-fold, 'convex-concave' ambiguity in reconstructing shape from shading. However, for the 'concave' solution, the occluding boundary is seen edge on in the image, which clearly constitutes an accidental alignment of the viewing direction with the rim of the surface. Moreover, this solution is impossible for a closed object—there is no way to extend the surface to a closed object without occluding it. The second solution is therefore excluded as an acceptable one.

These theorems may be summarized as stating that for a Bruss reflectance function, and an object wholly contained in the image, there is almost always a unique solution to shape from shading. In sketching their proof, we will assume for simplicity that the imaged object is actually Lambertian, and that the illumination is due to an infinitely distant point source. We will first derive constraints on the solutions to shape from shading which are valid for the case of general light source direction. (Note, however, that we exclude backlighting: the light source and viewer must be in the same hemisphere with respect to the imaged object.) Then, we will indicate how the proofs can be completed when the illumination is from the viewer direction.

With these simplifications, the image intensity is given

by

$$I = -\hat{n} \cdot \hat{L}, \quad (1)$$

where \hat{L} is the light ray direction, and \hat{n} is the surface normal. Explicitly,

$$\hat{n} \equiv \frac{(p, q, -1)}{(1 + p^2 + q^2)^{1/2}}.$$

The surface normal is defined so that it points towards the viewer located at negative z .

3 Characteristic Strips as Surface Curves

A *characteristic strip* is, roughly, a line in the image along which the surface depth and orientation can be computed, assuming that these quantities are known at the starting point of the line. Characteristic strips were used by Horn in his original algorithm for reconstructing shape from shading. (see, e.g., [5]). A consistent solution to shape from shading determines a *flow* of characteristic strips in the image, with every image point lying on exactly one characteristic strip line. Conversely, such a flow of characteristic strips uniquely determines a shape solution. Most of our proof is focused on uniquely specifying the flow of characteristic strips, and thus, from the above, a unique shape solution.

The characteristic strip curves are computed by solving a system of differential equations. In terms of a Hamiltonian function H [7], where

$$H(x, y, p, q) \equiv I(x, y) + \hat{n} \cdot \hat{L} = 0,$$

the characteristic strip equations can be written as:

$$\dot{x} = H_p \quad (\equiv \frac{\partial H}{\partial p}), \quad \dot{y} = H_q, \quad \dot{p} = -H_x, \quad \dot{q} = -H_y. \quad (2)$$

The dot denotes a derivative with respect to 'time', an arbitrarily chosen variable that parameterizes the position along the characteristic strip. The subscripts denote partial differentiation. Explicitly,

$$\dot{x} = H_p = \frac{L_x}{(1 + p^2 + q^2)^{1/2}} - p \frac{\hat{n} \cdot \hat{L}}{(1 + p^2 + q^2)},$$

$$\dot{y} = H_q = \frac{L_y}{(1 + p^2 + q^2)^{1/2}} - q \frac{\hat{n} \cdot \hat{L}}{(1 + p^2 + q^2)}.$$

Also,

$$\dot{z} = p\dot{x} + q\dot{y},$$

by the definition of p, q . After some algebra, and a redefinition of the arbitrary time parameter, the above equations may be rewritten as:

$$\frac{d\vec{r}}{dt} \equiv \frac{d}{dt} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \hat{L} - (\hat{n} \cdot \hat{L})\hat{n}. \quad (3)$$

A curve generated by this equation is on the surface of the illuminated object, and will be referred to as a *surface strip*. Note our convention for the time direction: as time increases, the trajectory heads away from the light source.

This form makes it clear that the surface curve corresponding to a characteristic trajectory depends only on the 3D object and the illumination direction, and not on the viewing direction. In other words, if the lighting is kept constant, characteristic strips in different images of an object represent the identical surface strips. Also, it is clear that the surface strips can be extended smoothly beyond the illuminated region in accordance with eq. 3, so as to cover the whole of the object. The right-hand side of this equation is just \hat{L} projected into the surface tangent plane. Thus, *surface strips are curves of steepest ascent in the \hat{L} direction.*

4 Singular Points and the Flow of Characteristic Strips

In this section, we derive properties of singular points, and describe the global structure of the flow of characteristic strips for a generic image. At singular points, the right-hand sides of the equations 2 and 3 vanish, so that a characteristic strip originating exactly at a singular point never leaves it. Singular points are therefore *fixed points* of the characteristic strip equations.

Also, singular points correspond to points on the object where the surface normal is parallel to the light source direction. Thus, considered as object points, they depend only on the light source direction, and not on the viewer. We have seen that surface strips are also independent of the viewer. Thus, it is clear that the flow of characteristic strips in the image can be analyzed by considering the flow of surface strips on the corresponding object.

Consider a smooth, closed surface S . Define a cartesian coordinate system (ξ_1, ξ_2, ξ_3) , where $\xi_3 = \hat{L}$, i.e., the third coordinate measures distance along the light ray direction, and consider ξ_3 as a function on S . A *critical point* of ξ_3 is a point on the surface where the derivative of ξ_3 vanishes. A critical point is *non-degenerate* if the matrix of second derivatives of ξ_3 is non-singular there. Thus, a non-degenerate critical point is a local maximum, minimum, or saddle point of ξ_3 ; the surface at this point is respectively concave, convex, or saddle-shaped.

A singular point in the image corresponds exactly to a critical point on the object. Also, corresponding to the three different types of non-degenerate critical points, one can show that the flow of characteristic strips near their singular point images can be classified into three possibilities. This follows from the Grobman-Hartman theorem [9], and was first pointed out by Saxberg [12]. A local minimum or convex surface corresponds to a *source* singular point, with an infinite number of outwards-pointing characteristic strips originating at the point. Similarly, a local maximum corresponds to a *sink* singular point, with all nearby strips converging. For a *saddle* singular point, the flow is *saddled*. Each saddle point is the origin of two characteristic strips, and similarly the term *source* and *sink* are used for two. Other characteristic strips approach the saddle point, without actually connecting to it.

ing to it (see, e.g., [7]).

In [8], the following theorem is proven. A closed surface S is defined to be *structurally stable* with respect to the function ξ_3 if 1) the surface has a finite number of critical points, 2) they are all non-degenerate, 3) there are no surface strips connecting saddle critical points. **Theorem:** 1) *Every smooth, closed surface can be approximated arbitrarily well by a structurally stable surface.* 2) *For all sufficiently small perturbations of a structurally stable surface, the perturbed surface is also structurally stable.* Thus, almost all surfaces are structurally stable.

It is easy to show that at a singular point corresponding to a non-degenerate critical point on the object, the second derivative matrix of the intensities is non-singular. A singular point with this property will also be referred to as *non-degenerate*. Then, for an image of a structurally stable object, the above results imply: 1) *the number of singular points in the image is finite*, 2) *every singular point is non-degenerate*, and 3) *there is no characteristic strip connecting two saddle-type singular points*. The last property is valid for the flow of strips corresponding to the structurally stable object. Eventually, we will define a *structurally stable image* to be one that has these three properties. However, to make this definition precise, one further result is needed. In any case, from the theorem, it is clear that structurally stable images are generic. From now on, we consider only these generic images and objects.

By taking advantage of the interplay between surfaces and their images, one can demonstrate many properties of surface and characteristic strips rather easily. For example, it is easy to show that all surface strips begin and end at critical points. Thus, all characteristic strips begin and end either at a singular point, or on the limb (if a terminal singular point is occluded). Also, by the Poincaré-Hopf index theorem of differential geometry (see, e.g., [1], [4]), one can show that:

$$N_{\text{ell}} - N_{\text{sad}} = E,$$

where N_{ell} is the number of critical points with positive curvature (local maxima or minima of ξ_3), N_{sad} is the number of saddle critical points, and E is the Euler number of the surface. For a genus zero surface, $E = 2$.

5 Sketch of the Uniqueness Proof

The first step in the proof is a local uniqueness theorem. It was previously shown by Bruss [2] that for some local neighborhood of a non-degenerate singular point in the image, there exists a *unique* surface solution that is convex around the singular point, and a unique surface solution that is concave around this point. She proved this for the case of Bruss reflectance functions, but it was later realized by Saxberg [12] that the result was valid for illumination from a general light source direction. In fact, this local result is the essential content of Bruss' uniqueness theorem.

In [7], it is proven that there are most two additional, saddle-shaped solutions in the local neighborhood of a non-degenerate singular point. So far, this result has

only been demonstrated for Bruss reflectance functions. It implies that there is at most a four-fold ambiguity in reconstructing shape locally around a non-degenerate point. The proof of this fact is difficult, and is based on the mathematical theory of dynamical systems. However, the result is actually not very important in the uniqueness proof, as we shall see. It is used only to give a precise definition to the class of structurally stable images. This point is elaborated in a brief digression.

The 'definition' of structurally stable images in the previous section was flawed in that it referred to the imaged object. Since we are only presented with the image, and have no knowledge of the object, the definition should be amended so that it only refers to the image. The third defining property is therefore altered. A structurally stable image has the defining property: 3) *For any two singular points in the image, consider the local saddle solutions around these points (if they exist). For any possible extension of these solutions, there is no characteristic strip which connects the two singular points.* This guarantees that, under any interpretation of the singular points, there will be no characteristic strip connecting saddle points in a structurally stable image. The proof that this amended definition yields a generic class of images relies on the above result that there are only a finite number of saddle solutions to consider. We now return to outlining the uniqueness proof.

The second step of the proof is the demonstration that the convex and concave solutions around singular points can be extended over large image regions (assuming a consistent solution exists). In fact, the image can essentially be covered by these regions. This does not quite determine the solution, since the relative depths of the singular points may not be known—one needs to determine how the splicing of the different solution regions should be done.

The third step proves that all singular points in a consistent solution actually are connected together by sequences of characteristic strips. Since the surface can be computed along these strips, this determines the relative depths of the singular points and the splicing of the different solution regions.

The above is sufficient to show that if the nature of the surface solution is known at each singular point—i.e., whether it is concave, convex, or saddle-shaped—then the shape solution is uniquely determined. The last step in the proof shows that the type of the solution at each singular point is in fact uniquely specified, and that the solution is therefore unique.

The steps of the proof are now described in more detail. The second result follows from well known properties of characteristic strips—basically, the standard existence theorems of differential equations. Also, it depends on the observation made earlier that every strip on a consistent surface begins and ends at a critical point, and on the Grobman-Hartman result that there are only four strips connecting to any saddle point. It is valid in the general illumination direction case.

The third step is also true for general illumination direction for the surface strips on the object. For char-

acteristic strips in the image—the important case—it is valid if the relevant connecting strips are not partially occluded. For Bruss reflectance functions, this partial occlusion never occurs. The complete proof of this step is given in [7]. Although it is somewhat technical, its essential content is a topological or continuity argument. Also, it is argued in [8] that the partial occlusion of connecting characteristic strips is probably not very significant even for general light source direction.

Various arguments are used to demonstrate the last step. One of the most important, which is valid also for general light source direction, is the following. Suppose some singular point is assumed to correspond to a source, that is, to a convex solution. This uniquely determines the characteristic strips emanating from this point in the image. Then, one can show that the nature of the solution at any other singular point to which the first is connected by a characteristic strip is *uniquely determined*. This again follows from a topological argument. By a chain reaction of this reasoning, one can determine the nature of the solutions at many singular point which are connected to the original source by sequences of strips. An equivalent result holds if the original singular point is a sink.

Another important constraint on the solution is due to the *shadow boundary*, defined as the curve separating the illuminated from the non-illuminated region of the object. For general light source direction, we assume that the object is *non-self-shadowing*, where this is defined similarly to our earlier definition of 'non-self-occlusion'. Then, by similar arguments to those following the statement of the uniqueness theorems in section 2, e.g., by non-accidentalness, the surface at the shadow boundary must be 'convex', and 'rolling away' from the light source. Since characteristic strip lines projected on the object are lines of steepest ascent in the direction away from the light source, this implies that these strips are *exiting only* at the shadow boundary. Thus, a singular point connected to the shadow boundary by a strip cannot be a sink, but must correspond to a convex or saddle-shaped surface.

These results are already enough to show that the solutions to shape from shading are strongly constrained for general light source direction. A more complete treatment is presented in [8]. For Bruss reflectance functions, the essential ingredient giving uniqueness is the fact that the *shadow boundary is completely visible in the image, and coincides with the boundary of the object as projected in the image*. The occluding boundary per se is not important. In this case, the strips at the image boundary exit only. As a consequence, surface strips connecting visible singular points are never occluded, as stated previously—if they were to exit the visible region of the object, they could not return to it. Another consequence is that the Poincaré-Hopf theorem quoted in section 4 can be applied, and determines uniquely the number of saddle type singular points in the image [7]. Finally, using some other topological arguments, the fourth and last step of the uniqueness theorem can be proven.

6 Non-Existence of Solutions

Characteristic strips corresponding to a consistent shape solution cannot intersect in the image. This is so because an intersection would imply two different surface orientations corresponding to a single image point. On the other hand, for an arbitrary intensity function which is not a true image, it is perfectly possible that the computed characteristic strips do intersect.

Suppose we begin with a true image produced with a Bruss reflectance function, in which some singular point is uniquely determined to be a source. This in turn uniquely determines the characteristic strips emanating from this point. In [7], it is argued that a small perturbation of the intensities can be found for which 1) the only consistent interpretation of the given singular point remains a source, and 2) some of the strips emanating from this source intersect due to the perturbation. Then, this perturbed image cannot correspond to a physical object—it is an 'impossible' image.

Clearly, an impossible image with intersecting strips will continue to have intersecting strips for a small enough perturbation of its intensities. Thus, impossible images are an open subset of all images.

7 Shape Ambiguities near the Occluding Boundary

In this section, the characteristic strip equations near the limb are examined, and it is shown that the limb does not constrain the shading solution. First, the equations are rewritten in a rotated coordinate system, in terms of variables that remain finite on the limb (see also [12]). The characteristic strip equations are eq. 3 above and:

$$\dot{p} = -\frac{\partial I}{\partial x}(1+p^2+q^2)^{1/2}, \quad \dot{q} = -\frac{\partial I}{\partial y}(1+p^2+q^2)^{1/2}, \quad (4)$$

with the surface represented by $z(x, y)$, p , q . Instead, we switch to representing the surface by a function $y(x, z)$, and y_x , y_z , which should be possible at least locally. These quantities indeed remain finite on the limb.

Define the variables $w \equiv -p/q$, and $v \equiv 1/q$. If a surface solution $y(x, z)$ exists, then $w = y_x$ and $v = y_z$. These variables evolve via:

$$\begin{aligned} \dot{w} &= \frac{-\dot{p}}{q} + \frac{p\dot{q}}{q^2} = \frac{v}{|v|}(I_x + wI_y)(1+w^2+v^2)^{1/2}, \\ \dot{v} &= -\frac{\dot{q}}{q^2} = \frac{v}{|v|}(vI_y)(1+w^2+v^2)^{1/2}. \end{aligned} \quad (5)$$

The existence of a surface solution $y(x, z)$ also implies that

$$I_x \equiv \frac{\partial I}{\partial x} \Big|_z = I_x + y_x I_y, \quad I_z \equiv \frac{\partial I}{\partial z} \Big|_x = y_z I_y,$$

where on the left-hand side $I(x, y = y(x, z))$ is thought of as a function of x, z . Although I_x or I_y must be singular on the limb, I_x and I_z are both *finite*. The equations 5 appear to have almost the same form in the rotated coordinated system as they did in the original one, with y playing the role of z .

At least one surface solution of the characteristic strip equations can be assumed to exist—namely the original object from which the intensity image was derived. The basic strategy of our proof is to write the equations with reference to this “pre-existing” solution. Below, therefore, y_x and y_z denote the partial derivatives corresponding to this reference solution. Similarly, I_x^r and I_z^r denote the partial derivatives of the intensity assuming the reference solution. These quantities should be thought of as fixed functions of x and y , like the intensity I itself. They have the nice property that they remain finite on the limb.

One has:

$$I_y = \frac{I_x^r}{y_z}, \quad I_x = I_x^r - I_z^r \frac{y_x}{y_z}.$$

Substituting into the eqs. 5 yields:

$$\begin{aligned} \dot{w} &= \text{sign}(v) \left(I_x^r + I_z^r \frac{w - y_x}{y_z} \right) (1 + w^2 + v^2)^{1/2}, \\ \dot{v} &= \text{sign}(v) v \frac{I_z^r}{y_z} (1 + w^2 + v^2)^{1/2}. \end{aligned} \quad (6)$$

The surface normal can be written as:

$$\hat{n} = \text{sign}(v) \frac{(-w, 1, -v)}{(1 + w^2 + v^2)^{1/2}}. \quad (7)$$

Since \hat{n} is expressed as a function of w and v , the system of equations 3 and 6 form a complete set that can be solved for x, y, z, w, v as functions of time. For the reference solution, the surface normal \hat{n}_x^r has the same form as eq. 7 with (v, w) replaced by (y_z, y_x) . Since the limb is defined by $n_x^r = 0$, it follows that $y_z = 0$ on the limb, and only on the limb. The singularity of the eqs. 6 is due solely to this vanishing of y_z .

Let us restrict our considerations to a small portion of the image bordering the limb. Image plane coordinates are chosen so that the limb is tangent to the x direction at some point; this point can be taken to be the origin. For convenience, we assume that y attains a local maximum at this point, so that the limb is convex there. Also, it is assumed that the reference surface is ‘rolling away’ at the limb.

In the evolution equation for v , the variation of v is proportional to v itself, so that v never changes sign except at singularity points of the right-hand side, namely on the limb. Therefore, v and y_z can be chosen positive over the given region, and the sign factors in eqs. 6 and in \hat{n} can be neglected. Henceforth, we will drop the sign factors from these equations. Also, the sign of I_z^r is assumed negative over the given region. From the equation for v in eq. 6, this implies that v decreases with increasing time, and similarly for y_z in the reference solution. This choice of sign therefore implies that the direction of the characteristic strips is outwards at the limb, towards the invisible region. The singularity in the eqs. 6 implies that v and w are forced respectively towards 0 and y_x as a strip approaches the limb.

The possibility of multiple solutions to shape from shading arises from the following fact: although v and $w - y_x$ scale towards zero with y_z near the limb, their

absolute scale is not determined. Thus, we again define new variables

$$s \equiv \frac{v}{y_z}, \quad r \equiv \frac{w - y_x}{y_z}. \quad (8)$$

The time evolution equation for s is:

$$\dot{s} = \frac{s}{y_z} (I_z^r D^{1/2} - \dot{y}_z), \quad (9)$$

where

$$D \equiv 1 + w^2 + v^2 = 1 + y_z^2 s^2 + (y_z r + y_x)^2. \quad (10)$$

Since

$$\left. \frac{\partial}{\partial x} \right|_y = \left. \frac{\partial}{\partial x} \right|_z - \frac{y_x}{y_z} \left. \frac{\partial}{\partial z} \right|_x, \quad \left. \frac{\partial}{\partial y} \right|_x = \frac{1}{y_z} \left. \frac{\partial}{\partial z} \right|_x,$$

the time evolution of y_z is:

$$\dot{y}_z = \dot{x}(y_{zx} - \frac{y_x}{y_z} y_{zz}) + \dot{y} \frac{y_{zz}}{y_z}.$$

From the equation for I , eq. 1,

$$I_z^r = \frac{(y_{xz}, 0, y_{zz}) \cdot \hat{L}}{D_r^{1/2}} - \frac{I}{D_r} (y_x y_{xz} + y_z y_{zz}), \quad (11)$$

where

$$D_r \equiv 1 + y_z^2 + y_x^2.$$

After some algebra, one obtains an expression for the time evolution of s which is clearly *well defined* at the limb [8]. The evolution depends on the first and second derivatives of y for the reference solution, on I , and on r and s . The equation for r can be derived similarly, and is also well defined on the limb. With s and r finite and $y_z = 0$ on the limb, $w = y_x$ and $v = y_z = 0$ are forced there, as stated above.

y_z is not a continuously differentiable function of the image plane coordinates x and y . Thus we replace the variable y by $z_r(x, y)$, the value of z for the reference solution given x, y . The differential equation for the time evolution of z_r is:

$$\begin{aligned} \dot{z}_r &= - \left(\frac{y_x}{y_z} \right) \left(L_x - \frac{I}{D^{1/2}} (r y_z + y_x) \right) + \frac{1}{y_z} \left(L_y + \frac{I}{D^{1/2}} \right) \\ &= L_x + \frac{y_x I r}{D^{1/2}} + \frac{I}{y_z D^{1/2}} (-D_r^{1/2} D^{1/2} + 1 + y_x^2). \end{aligned} \quad (12)$$

Again, it is clear that this is well defined on the limb when $y_z = 0$. The system of equations for $\mathbf{P} \equiv (x, z_r, s, r)$ is consistent and *differentiable everywhere including on the limb*. Let the limb considered as a curve in the x - z_r plane be parameterized by σ . For arbitrary choice of initial conditions for s, r on the limb, the usual theorems of differential equations state that there exists a unique solution of the system of differential equations, with \mathbf{P} a differentiable function of σ and t . Given this solution, y and z can also be computed by simple integration of their equations of motion.

It now remains to show that $w = y_x, v = y_z$, and that the parametric form expressing x, y, z in terms of t and σ can be converted into a surface function $y(x, z)$ or

$z(x, y)$. This can be done via a mostly standard argument [3] [8]. The conclusion is that there is a range of possible solutions specified by 1) the choice of $z(x, y)$ on the limb, and 2) by the choice of a combination of the parameters $s(x, y)$ and $r(x, y)$ on this boundary. Moreover, these surface solutions are non-self-occluding. The shape solution in the neighborhood of the limb therefore has a *two parameter ambiguity*, as opposed to the one parameter ambiguity around an interior image curve [3]. It is also possible to generate arbitrary numbers of solutions by specifying the depth on an interior image curve that terminates at the limb [8]. This fact is used for the example image of the next section.

8 Shape from Shading as a Partially Ill-Posed Problem

In this section, an image example is presented for which shape from shading is ill-posed over a (very) small image region. Over the bulk of the image, the shape is uniquely determined. The exceptional region consists of points which are unconnected by characteristic strips to the only singular point in the image—a source. The region is bounded by the limb and by part of a strip from the source: the bounding strip at one point is tangent to the limb, but actually exits the image elsewhere. Thus, this region is completely isolated from the source. Since we have argued that only singular points constrain the shape solution, and not the limb, shape reconstruction is expected to be ambiguous in this region. This has been demonstrated numerically in [8]. It is also argued there that such 'ill-posed regions' are expected to be small fractions of the image.

The imaged object is essentially an elongated egg-shape, viewed from a direction reasonably distinct from the long axis of the egg. In a body-centered coordinate system, it is given by:

$$x_b^2 + y_b^2 + \frac{|z_b|^3}{27} = 1. \quad (13)$$

In this coordinate system, the light source direction is:

$$\hat{L}_b = (1, -1, 5), \quad (14)$$

and the viewing direction is:

$$\tilde{v}_b = (-\sin, 0, \cos), \quad \sin = .15, \quad \cos = (1 - \sin^2)^{1/2}. \quad (15)$$

The object is considered to be at positive z with respect to the viewer.

Our strategy for generating new solutions corresponding to the ill-posed image region is as follows. We choose an initial curve spanning the region, which runs from the bounding strip to the limb. The surface depths are then perturbed essentially arbitrarily on this curve, except that the perturbed surface must join smoothly onto the standard one at the bounding strip. Finally, by computing numerically the characteristic strips passing transversely through this curve, the perturbed solution is extended over the whole ill-posed region. The result for one arbitrary perturbation is shown in Figure 1, from a viewpoint behind the object, i.e., opposite the viewer.

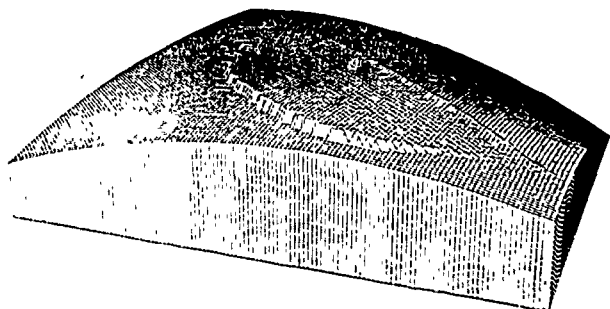


Figure 1: A composite view of the standard and perturbed surfaces, showing the discrepancy between them.

This is a composite picture, with the standard and perturbed solutions displayed together. That the two solutions differ is clear due to the sharp discontinuity between the occluding boundaries of the two surfaces.

9 Recovering Shape Locally from Image Intensities

In this section, it is shown that if a surface is locally quadratic at a point p , then it can be recovered locally directly from the image data. Special cases (e.g. singular points, or equal principal curvatures) are not considered here, and can be handled separately. The surface is assumed Lambertian, and the illumination is by an infinitely distant point source. For local shape reconstruction, these are reasonable assumptions apart from specularities.

The stipulation that the surface be locally quadratic means that there is a coordinate system at p in which the surface can be represented by:

$$z = \frac{1}{2}(a\underline{x}^2 + b\underline{y}^2). \quad (16)$$

This coordinate system, distinguished by underlined coordinates, will be referred to as the *object coordinate system*. Its origin is p . At p , the unit vector \hat{z} is equal to the surface normal \hat{n} (note that this differs from the convention of previous sections), while \hat{x} and \hat{y} give the principle curvature directions. At points away from p , the surface normal is:

$$\hat{n} = \begin{pmatrix} -a\underline{x} \\ -b\underline{y} \\ 1 \end{pmatrix} \frac{1}{(1 + a^2\underline{x}^2 + b^2\underline{y}^2)^{1/2}}. \quad (17)$$

Differentiating the image irradiance equation in this coordinate system yields:

$$I_{\underline{x}} = \hat{L} \cdot \begin{pmatrix} -a \\ 0 \\ 0 \end{pmatrix} \frac{1}{(1 + a^2 \underline{x}^2 + b^2 \underline{y}^2)^{1/2}} - \frac{I a^2 \underline{x}}{1 + a^2 \underline{x}^2 + b^2 \underline{y}^2}, \quad (18)$$

with a similar equation for $I_{\underline{y}}$. (Although the intensity I is a function of the image plane coordinates, it can be considered also as a function of \underline{x} and \underline{y} through parameterizing the surface with these variables, as detailed below.) At p , the above equations reduce to:

$$I_{\underline{x}} = \hat{L} \cdot \begin{pmatrix} -a \\ 0 \\ 0 \end{pmatrix}, \quad I_{\underline{y}} = \hat{L} \cdot \begin{pmatrix} 0 \\ -b \\ 0 \end{pmatrix}. \quad (19)$$

Differentiating again yields at p :

$$I_{\underline{x}\underline{x}} = -a^2 I, \quad I_{\underline{x}\underline{y}} = 0, \quad I_{\underline{y}\underline{y}} = -b^2 I.$$

These equations are not directly useful since they are expressed in terms of the unknown object coordinate system: they must be converted into the camera system. Let the origin of the camera coordinate system be at p also. Then the camera coordinate system is related to the object coordinate system by a rotation:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = R \begin{pmatrix} \underline{x} \\ \underline{y} \\ z(\underline{x}, \underline{y}) \end{pmatrix}. \quad (20)$$

Since \underline{z} is a function of \underline{x} and \underline{y} , the image plane coordinates x and y can be considered functions of these two variables also, via the above equation.

In a unified notation one has:

$$\frac{\partial}{\partial r_i} = \sum_j \frac{\partial r_j}{\partial r_i} \frac{\partial}{\partial r_j},$$

where the subscripts i, j denote the x or y coordinates, i.e., $(r_x, r_y) \equiv (x, y)$. From eqs. 20 and 16,

$$\frac{\partial}{\partial r_i} = \sum_j (R_{ji} \frac{\partial}{\partial r_j} + c_i r_i R_{jz} \frac{\partial}{\partial r_j}), \quad (21)$$

where $(c_x, c_y) \equiv (a, b)$.

The first derivative equations at p , eq. 19, can be rewritten as:

$$\sum_j R_{ji} \frac{\partial I}{\partial r_j} = -c_i \hat{L}.$$

This corresponds to two equations, for $i = \underline{x}$ and \underline{y} .

The second derivatives evaluated at p are:

$$\frac{\partial^2 I}{\partial r_i \partial r_j} = \sum_{km} R_{ki} R_{mj} \frac{\partial^2 I}{\partial r_k \partial r_m} + c_j \sum_{km} R_{ki} R_{mz} \frac{\partial r_j}{\partial r_k} \frac{\partial I}{\partial r_m}.$$

The expression

$$\sum_k R_{ki} \frac{\partial r_j}{\partial r_k},$$

which appears in this equation must be evaluated at p . This can be done since $\frac{\partial r_k}{\partial r_j} = R_{kj}$ at p , and:

$$\sum_k R_{ki} \frac{\partial r_j}{\partial r_k} = \sum_k \frac{\partial r_k}{\partial r_i} \frac{\partial r_j}{\partial r_k} = \delta_{ji}.$$

Substituting this result into the equation for the second derivatives yields:

$$\frac{\partial^2 I}{\partial r_i \partial r_j} = \sum_{km} R_{ki} R_{mj} \frac{\partial^2 I}{\partial r_k \partial r_m} + \delta_{ij} c_i \sum_m R_{mz} \frac{\partial I}{\partial r_m}.$$

Now, define three unit vectors to be the columns of the rotation matrix R :

$$\begin{pmatrix} \hat{R}_x & \hat{R}_y & \hat{R}_z \end{pmatrix} \equiv R.$$

Further, define:

$$\vec{A} \equiv \begin{pmatrix} I_x \\ I_y \\ 0 \end{pmatrix},$$

$$T \equiv \begin{pmatrix} I_{xx} & I_{xy} & 0 \\ I_{xy} & I_{yy} & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

In terms of these quantities, and expressed in the camera coordinate system, the image irradiance equation and the derivative equations are:

$$I = \hat{L} \cdot \hat{R}_z, \quad (22)$$

$$(\vec{A} + a \hat{L}) \cdot \hat{R}_x = 0, \quad (23)$$

$$(\vec{A} + b \hat{L}) \cdot \hat{R}_y = 0, \quad (24)$$

$$\hat{R}_x^T T \hat{R}_x + a \vec{A} \cdot \hat{R}_x + a^2 I = 0, \quad (25)$$

$$\hat{R}_y^T T \hat{R}_y + b \vec{A} \cdot \hat{R}_y + b^2 I = 0, \quad (26)$$

$$\hat{R}_z^T T \hat{R}_z = 0, \quad (27)$$

These are the six basic equation that we will work with. The unknowns to be determined include the direction of the surface normal \hat{n} (two unknowns), the direction of maximal curvature of the object surface (one unknown, assuming \hat{n} is known), and the magnitudes of the principle curvatures of the object, a and b . Thus there are a total of five unknowns, and the above six equations are sufficient to determine these, and also provide a measure of the error when there is no exact solution, for instance because of departures from the surface model. (This fails in special cases which can be handled separately.) Alternatively, if the surface albedo is also unknown, in principle it too could be recovered.

The solution of these equations is carried out as follows. For convenience, the camera coordinate system is defined so that the light source direction \hat{L} is in the x - z plane. The first of the six equations constrains \hat{R}_z . In terms of variables r and s satisfying

$$s \equiv \pm \sqrt{1 - I^2 - r^2},$$

\hat{R}_z may be written as:

$$\hat{R}_z = \begin{pmatrix} L_x r + L_x I \\ s \\ -L_x r + L_x I \end{pmatrix}.$$

The strategy in solving the six equations will be first to express all unknowns in terms of r , and then to solve for r .

Since $\hat{R}_x \cdot \hat{R}_x = 0$ by the orthogonality of R , and from eq. 23, one has:

$$\hat{R}_x \sim \hat{R}_x \times (\vec{A} + a\hat{L}). \quad (28)$$

Thus \hat{R}_x is specified, up to an unimportant sign, in terms of \hat{R}_x (i.e. r), a , b . A similarly result holds for \hat{R}_y . The orthogonality relation $\hat{R}_y \cdot \hat{R}_x = 0$ leads to:

$$\vec{A} \cdot \vec{A} - (\hat{R}_x \cdot \vec{A})^2 + (a+b)(\vec{A} \cdot \hat{L} - I(\hat{R}_x \cdot \vec{A})) + ab(1 - I^2) = 0, \quad (29)$$

where eq. 22, and the fact that a unit vector has unit norm, have been used to simplify expressions.

Also, eq. 27 can be written as:

$$(\hat{R}_x \times (\vec{A} + a\hat{L}))^T T(\hat{R}_x \times (\vec{A} + b\hat{L})) = 0. \quad (30)$$

As in eq. 29, this can be written as a linear combination of $a + b$ and ab .

Eqs. 30 and 29 are two linear equations for the two unknowns $(a + b)$ and ab , with coefficients depending on known constants and the variable r . Thus the principle curvatures a , b can be solved for in terms of r . If for convenience we define a to be larger than b , then the solution is unique. Note that there is no convex-concave-saddle ambiguity, as there would be for p a singular point.

a , b , and therefore \hat{R}_x , \hat{R}_y , are now expressed in terms of r . Substituting for these quantities in eqs. 25 and 26 yields two complicated equations for r , which can be converted into high order polynomial equations. r can therefore be determined as the root of a high order polynomial. However, the only admissible solutions for r are those which satisfy the condition

$$|r| \leq (1 - I^2)^{1/2},$$

following from the fact that \hat{R}_x has unit norm. Therefore the strategy we adopt for finding the solutions r is a simple search: the above admissible range of values for r is sampled, in order to bracket the zeros of eqs. 25 and 26. This procedure has been implemented, and works reasonably well. Moreover, in all cases studied so far, there is a unique solution for r when a common solution to both equations exists. Thus the image data appears to determine a unique local solution for the object surface shape, assuming that this surface is quadratic.

Acknowledgments

I would like to acknowledge useful and enjoyable conversations with B. Saxberg, Don O'Shea, B. K. P. Horn, and Richard Weiss.

References

1. V. I. Arnold, *Ordinary Differential Equations*. MIT Press: Cambridge, MA, 1973.
2. A. R. Bruss, "The Eikonal Equation: Some Results Applicable to Computer Vision," *Journal of Math. Phys.* 23(5): 890-896, May 1982.
3. P. R. Garabedian, *Partial Differential Equations*. John Wiley: New York, N. Y., 1969.

4. V. Guillemin and A. Pollack, *Differential Topology*. Prentice-Hall: New Jersey, 1974.
5. B.K.P. Horn and M.J. Brooks (eds.) *Shape from Shading*. MIT Press: Cambridge, MA, 1989.
6. F. P. Ferrie and M. D. Levine, "Where and Why Local Shading Analysis Works," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 2, pp. 198-205, 1986.
7. J. Oliensis, "Existence and Uniqueness in Shape From Shading," University of Massachusetts TR 89-109, October 1989, and in the *Tenth International Conference on Pattern Recognition*, Atlantic City, New Jersey, June 17-21, 1990, pp. 341-345.
8. J. Oliensis, "Shape from Shading as a Partially Ill-Posed Problem," University of Massachusetts TR 90-50, June 1990.
9. J. Palis and W. de Melo, *Geometric Theory of Dynamical Systems*. Springer-Verlag: NY, 1982.
10. A.P. Pentland, "Local Shading Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 6, No. 2, pp. 170-187, March 1984.
11. B. Saxberg, "An Application of Dynamical Systems Theory to Shape From Shading," in Proc. DARPA Image Understanding Workshop, Palo Alto, CA, May 1989, pp. 1089-1104.
12. B. V. H. Saxberg, "A Modern Differential Geometric Approach to Shape from Shading," MIT Artificial Intelligence Laboratory, TR 1117, 1989.
13. R. Szeliski and B. K. P. Horn, "An Impossible Shaded Image," unpublished.
14. P. Giblin and R. Weiss, "Reconstruction of Surfaces from Profiles," in Proc. International Conference on Computer Vision, London, England, June 1987, pp. 136-144.

The Mathematical Foundations of Smoothness Constraints: A New Class of Coupled Constraints

M. A. Snyder

Computer and Information Sciences Department
University of Massachusetts at Amherst

June 28, 1990

Abstract

Gradient-based approaches to the computation of optical flow often use a minimization technique incorporating a smoothness constraint on the optical flow field. Smoothness constraints are also of interest in surface interpolation, where they are known as "performance functions." All known smoothness constraints used to compute optical flow have a subtle property, namely that they do not mix derivatives of different components of the optical flow field. We present an analysis of smoothness constraints which do not satisfy this "decoupled" property, but rather in which derivatives of different components of the flow can interact. By using the representation theory of the group of Euclidean motions in the image plane, we show that the single assumption that the smoothness constraint is invariant under this group of transformations allows us to write down a *complete* list of all possible invariant smoothness constraints of type (p, q) , by which it is meant that they are quadratic in p^{th} derivatives of the optical flow field, and in q^{th} derivatives of the grey level image intensity function. This is done explicitly for the values $0 \leq p, q \leq 2$. So far as the author is aware, all of these smoothness constraints, excepting those linear combinations which are decoupled, are new. We find that there are 4 of type $(1, 0)$, 5 of type $(2, 0)$, 8 of type $(1, 1)$, 14 of type $(1, 2)$, 15 of type $(2, 1)$, and 24 of type $(2, 2)$. In addition, we find using our method all invariant "performance measures," used in surface interpolation, when the performance measure is quadratic in no higher than fourth derivatives of the objective function.

1 Introduction

In this work we consider smoothness constraints which can be written as an integral, over the image plane, of a quantity called the smoothness density Σ (see [Snyd89] for a fuller discussion). We consider only densities which are quadratic in both p^{th} derivatives of the optical flow field

$\mathbf{U} = (U_1, U_2)^T \equiv (u, v)$, and in q^{th} derivatives of the grey-level image intensity function I . These smoothness densities are therefore of the form:

$$\Sigma = f_{c_1, \dots, c_q; d_1, \dots, d_q}^{a_1, \dots, a_p; b_1, \dots, b_p} (\partial_{a_1} \dots \partial_{a_p} U_r) (\partial_{b_1} \dots \partial_{b_p} U_s) I_{c_1 \dots c_q} I_{d_1 \dots d_q}, \quad (1)$$

where the quantity f_{\dots} does not involve derivatives of either \mathbf{U} or of I . As in our previous work [Snyd89], we are using the Einstein summation convention, in which all repeated indices are understood to be summed over from 1 to 2. We are using the notation that $\partial_k \equiv \partial/\partial x_k$ ($k = 1, 2$), and a subscript of I denotes differentiation with respect to that coordinate, e.g.,

$$I_{d_1 \dots d_q} = \frac{\partial^q I}{\partial x_{d_1} \dots \partial x_{d_q}}. \quad (2)$$

(Note that the subscripts of \mathbf{U} denote components, not derivatives.) We will call a smoothness density of the form (1) a *smoothness density of type (p, q)* . The well known smoothness constraint of Horn and Schunck [Horn81], for instance, is a particular smoothness constraint of type $(1, 0)$, while that of Nagel and Enkelmann [Nage86] is of type $(1, 2)$.

In our previous work [Snyd89], we asserted that a smoothness density should satisfy three physically reasonable conditions:

1. The smoothness density should be invariant under a change in the Cartesian coordinate system of the image, i.e., it should be invariant under the Euclidean group of the plane $\text{ISO}(2)$. We called this the 0^{th} Law of Computer Vision.
2. The smoothness density should be positive definite.
3. The smoothness density should be decoupled, by which is meant that no terms involving the product of derivatives of different components of \mathbf{U} (e.g., $u_x v_y$) should appear in (1).

We used these three constraints in [Snyd89] to give a completeness proof for decoupled smoothness constraints

of type (1,1) and (1,2), i.e., we found all the independent such smoothness constraints.

It is the purpose of the present work to drop the third assumption, and hence to find *all* possible smoothness densities which satisfy the first two assumptions, namely Euclidean invariance and positive definiteness. We recall that the "decoupling" assumption in [Snyd89] had no obvious physical or mathematical motivation, but was characteristic of all known smoothness densities.

We cannot yet justify the *experimental* relevance of considering coupled smoothness constraints, but it is possible that coupled smoothness constraints may find future application in the analysis of visual motion. Our own interest in this problem, however, is from a purely theoretical standpoint.

We will use the notation and ideas developed in [Snyd89], to which the interested reader is referred for mathematical background and the physical justification of our approach.

We recall that an element of the Euclidean group ISO(2) of the plane is specified by a rotation matrix $R \in SO(2)$ and a translation vector t , and that this element (R, t) of the Euclidean group has the following effect on the position vector r of a point with respect to some origin.

$$(R, t) : r \longrightarrow r' = Rr + t. \quad (3)$$

Furthermore, an object $F_{k_1 \dots k_r}$ which transforms under this Euclidean transformation according to the rule:

$$F_{k_1 \dots k_r}(r) \longrightarrow F'_{k_1 \dots k_r}(r') = R_{k_1 k'_1} \dots R_{k_r k'_r} F_{k'_1 \dots k'_r}(r), \quad (4)$$

where r and r' are related by (3), is called an r^{th} rank tensor (under ISO(2)). An "ordinary" vector A (for example, the optical flow vector U) is just a 1st rank tensor, i.e., $A \longrightarrow A' = RA$.

It follows easily from the tensor transformation law that if Σ is to be an invariant under the Euclidean group, then the quantity f_{\dots} must be a tensor of rank $2p + 2q + 2$. In [Snyd90a] we show that the number of smoothness constraints of type (p, q) is in principle at most $(p + q + 2)(2p + 2q + 1)!!$. Of course, not all of these can be independent. In the table, we give the number of such tensors in principle, the maximum number which can be independent, and in the last column, the actual number of independent such tensors found in this work.

Clearly, the calculation of all the possible invariant smoothness constraints rapidly becomes a daunting task if one proceeds by trying to write down all possible tensors, even though various symmetries lead to relations between the tensors. We will not pursue this train of thought further.

In the next section, we develop the representation theory of ISO(2) (or, since all our quantities are manifestly translation invariant, the representation theory of SO(2)). In Section 3, we use these results to find simply and quickly *all* possible ISO(2)-invariant smoothness

constraints of type (1,0), (2,0), (0,1), and (0,2). We reproduce the well known decoupled smoothness constraints of these four types, and exhibit new coupled smoothness constraints. In Section 4, we use the results of Section 3 to find all invariant smoothness constraints of type (1,1) and (1,2). We reproduce the well known decoupled constraints of Nagel and Enkelmann [Nage83, Nage86] and give new coupled constraints. In Section 5, we find all the constraints of type (2,1) and (2,2), all of which, so far as we are aware, are new. In Section 6, we consider the case of performance functions for surface interpolation, and find all the invariant smoothness functions quadratic in 3rd and 4th derivatives. These are, so far as we are aware, new. We conclude with some remarks on further investigations of these ideas.

2 The Representation Theory of ISO(2)

Because all the quantities which appear in our smoothness density are translationally invariant, we need only consider the SO(2) subgroup of ISO(2). The reader unfamiliar with SO(2) can consult [Snyd89] for the necessary mathematical background. A general reference for this section is the book by Murnaghan [Murn38].

It is well known that the irreducible representations of SO(2) are labelled by an integer n , called the *weight* of the corresponding irreducible representation. We will denote this irreducible representation by n . In what follows we will use superscripts and subscripts to label where a particular representation comes from, but these indices have no other significance. Since SO(2) is an Abelian group, all its irreducible representations are one dimensional. The carrier space for such an irreducible representation can be taken to be the image plane, so that we say that a "vector" (not to be confused with an ordinary vector) $A = (A_1, A_2)^T$ transforms according to the irreducible representation (irrep) n if, under a rotation (of the coordinate system) by θ , A transforms like

$$A \longrightarrow A' = R(n\theta)A, \quad (5)$$

Type	Example	Number of Tensors	Maximum # Independent	Actual # Independent
(1,0)	$(\partial U)^2$	9	9	4
(2,0)	$(\partial^2 U)^2$	60	21	5
(1,1)	$(\partial U)^2 (\partial I)^2$	60	30	8
(1,2)	$(\partial U)^2 (\partial^2 I)^2$	525	60	14
(2,1)	$(\partial^2 U)^2 (\partial I)^2$	525	64	15
(2,2)	$(\partial^2 U)^2 (\partial^2 I)^2$	5,670	126	24
(0,1)	$(\partial I)^2$	9	3	1
(0,2)	$(\partial^2 I)^2$	60	6	2
(0,3)	$(\partial^3 I)^2$	525	10	2
(0,4)	$(\partial^4 I)^2$	5,670	15	3

where

$$R(n\theta) = \begin{pmatrix} \cos n\theta & \sin n\theta \\ -\sin n\theta & \cos n\theta \end{pmatrix}. \quad (6)$$

We write the fact that A transforms according to (5) as $A \sim n$. We note that if $n = 1$, then A is an "ordinary" vector (corresponding to the irrep 1).

It can be shown [Snyd89] that with no loss of generality, we can confine our considerations to non-negative weights. We note that an alternative, and somewhat more powerful, approach to the representation theory of $SO(2)$ is to associate with each vector $A = (A_1, A_2)^T$ the complex number $A = A_1 + iA_2$. The transformation law then assumes the simple form $A \rightarrow e^{in\theta} A$. We do not pursue this avenue here¹, although the complex approach is used extensively in the sequel to this work [Snyd90b].

In our work, we are interested in quantities of the form

$$A_{a_1}^{(1)} A_{a_2}^{(2)} \dots A_{a_k}^{(k)}, \quad (7)$$

where $A^{(i)}$ is a vector of weight n_i (normally, $n_i = 1$). Formally, (7) is a tensor. Such tensors are not, in general, irreps of $SO(2)$, but rather have "pieces" which do transform irreducibly under $SO(2)$, i.e., like (5) for some n . That is, a quantity like (7) can be written as a sum of quantities which transform irreducibly under $SO(2)$. In the language of group representation theory, we say that the representation of $SO(2)$ given by (7) is *reducible* under the action of $SO(2)$ into a direct sum of irreps of $SO(2)$.

These comments can be made more precise. A quantity of the form (7) is an element of the direct product space $V_2 \otimes V_2 \otimes \dots \otimes V_2$ (p factors of V_2), where V_2 is the image plane. It therefore transforms (following (7) according to the direct product representation

$$n_1 \otimes n_2 \otimes \dots \otimes n_k \quad (8)$$

of $SO(2)$. This transformation is not, in general, irreducible, but can be written as the direct sum of irreps of $SO(2)$. Our interest here is in the specific combinations of the $A_{a_i}^{(i)}$ that transform as *scalars* (invariants) under $SO(2)$, since these will correspond to $ISO(2)$ -invariant smoothness constraints. That is, we are interested in the irreps 0 contained in the direct sum decomposition of the direct product (8). This can be done as follows.

We first note that the direct product and direct sum of representations m , n , and p satisfy the following two rules:

$$\begin{aligned} m \otimes (n \oplus p) &= (m \otimes n) \oplus (m \otimes p) \\ m \otimes (n \otimes p) &= (m \otimes n) \otimes p. \end{aligned}$$

Consequently the reduction of (8) into a direct sum of irreps can be made if we can reduce the direct product $m_1 \otimes m_2$ of two irreps m_1 and m_2 .

¹The reader can easily show that if $A = (A_1, A_2)$ is a vector of weight 1, then the complex number $A = A_1 + iA_2$ transforms like $A \rightarrow A' = \exp i\theta A$, and hence, $(\Re A^n, \Im A^n)^T$ is a vector of weight n , where $\Re Z$ and $\Im Z$ are the real and complex parts of the complex number Z , respectively.

To see how this is done, consider a vector A of weight $m_1 \neq 0$, and a vector B of weight $m_2 \neq m_1 \neq 0$. Then the direct product representation $m_1 \otimes m_2$ consists of the four quantities

$$\{A_a B_b; a, b = 1, 2\} = \{A_1 B_1, A_1 B_2, A_2 B_1, A_2 B_2\}.$$

It is shown in [Snyd90a] that by taking the linear combinations of $\{A_a B_b\}$ given by

$$\begin{aligned} \Psi_1 &= A_1 B_1 + A_2 B_2 = B \cdot A, \\ \Psi_2 &= A_2 B_1 - A_1 B_2 = B \times A \\ \Phi_1 &= A_1 B_1 - A_2 B_2, \quad \Phi_2 = A_2 B_1 + A_1 B_2, \end{aligned} \quad (9)$$

then the following two vectors transform irreducibly under $SO(2)$:

$$\Psi = \begin{pmatrix} \Psi_1 \\ \Psi_2 \end{pmatrix} \rightarrow R((m_1 - m_2))\Psi \quad (11)$$

$$\Phi = \begin{pmatrix} \Phi_1 \\ \Phi_2 \end{pmatrix} \rightarrow R((m_1 + m_2))\Phi. \quad (12)$$

That is, Ψ is a vector of weight $m_1 - m_2$, so that $\Psi \sim m_1 - m_2$ and Φ is a vector of weight $m_1 + m_2$, so that $\Phi \sim m_1 + m_2$. (Note that $m \pm n$ is *not* any sort of "sum" or "difference" of two irreps—it is the representation of weight $m \pm n$.) This is exactly the decomposition of the direct product into a direct sum of irreps:

$$m_1 \otimes m_2 = m_1 - m_2 \oplus m_1 + m_2 \quad (m_1 \neq m_2 \neq 0). \quad (13)$$

If either m_1 or m_2 is zero, then the above derivation is invalid, since there are not four independent quantities of the form $\{A_a B_b\}$. In that case, it is clear that a scalar (~ 0) times a vector of weight m just gives back a vector of weight m , i.e., $m \otimes 0 = m$. A slight complication does arise, however, if $m_1 = m_2$. In that case (11) says that $\Psi_1 = A_1 B_1 + A_2 B_2 = B \cdot A$ and $\Psi_2 = A_2 B_1 - A_1 B_2 = B \times A$ are *both* invariants (note that in two dimensions, the cross product of two vectors which transform in the same way under $SO(2)$ is a scalar, not a vector). Furthermore, if $m_1 = m_2$, and both of these irreps come from the same quantity (i.e., $A = B$), then $\Psi_2 = 0$, so there is only one non-trivial invariant instead of two. In the latter case, we say that the representations m_1 and m_2 are *identical*. We summarize these results as:

$$m \otimes m = 0^+ \oplus 0^- \oplus 2m \quad (A \neq B, m \neq 0) \quad (14)$$

$$\begin{aligned} m \otimes m &= 0 \oplus 2m \equiv \\ &\equiv (m \otimes m)_{id} \quad (A = B, m \neq 0) \end{aligned} \quad (15)$$

$$m \otimes 0 = 0 \otimes m = m. \quad (16)$$

Here, 0^+ and 0^- are the appropriate Ψ_1 and Ψ_2 , respectively, and $2m$ is the representation of weight $2m$. We have introduced the notation $(m \otimes m)_{id}$ to denote the direct product of the two representations m when the representations are in fact identical.

These results will be crucial to the remainder of the analysis.

3 The Direct Sum Decomposition of $1^{\otimes m}$

The problem of finding all (coupled or decoupled) ISO(2)-invariant smoothness constraints of type (p, q) is just the problem of finding all possible linear combinations of quantities like

$$(\partial_{a_1} \dots \partial_{a_p} U_r) (\partial_{b_1} \dots \partial_{b_p} U_s) (I_{c_1 \dots c_q} I_{d_1 \dots d_q}),$$

such that the linear combinations transform as scalars under ISO(2) (this is exactly the point of finding all the tensors f_{\dots} !). But the set of all quantities of the form (3) spans the tensor product space $V_2 \otimes V_2 \otimes \dots \otimes V_2$ ($2p + 2q + 2$ factors of V_2). Since $\{\partial_a\}$ and $\{U_a\}$ transform according to the 1 representation of ISO(2), and I is a scalar, so that $\{\partial_a I\}$ transforms according to the 1 representation as well, it follows that

$$\begin{aligned} \partial_{a_1} \dots \partial_{a_p} U_r &\sim \underbrace{1 \otimes \dots \otimes 1}_{p+1} \equiv 1^{\otimes(p+1)} \\ I_{c_1 \dots c_q} &\sim \underbrace{1 \otimes \dots \otimes 1}_q \equiv 1^{\otimes q}. \end{aligned} \quad (17)$$

Consequently, (3) must transform according to the (reducible) representation

$$\left[1^{(p,0)} \right] \otimes \left[1^{(0,q)} \right], \quad (19)$$

of ISO(2), where we have defined:

$$\begin{aligned} (\partial_{a_1} \dots \partial_{a_p} U_r) (\partial_{b_1} \dots \partial_{b_p} U_s) &\sim 1^{\otimes(p+1)} \otimes 1^{\otimes(p+1)} \equiv 1^{(p,0)} \\ I_{c_1 \dots c_q} I_{d_1 \dots d_q} &\sim 1^{\otimes q} \otimes 1^{\otimes q} \equiv 1^{(0,q)}. \end{aligned}$$

Therefore, there is a one-to-one correspondence between invariants of the form (1), and representations of weight 0 which occur in the direct sum decomposition of (19) into irreps. *Therefore, we can find all the invariants of type (p, q) by simply making this decomposition.* Indeed, the decomposition will tell us not only how many invariants there are, but also gives us their explicit form, by using (9)! That is, the theory of group representations gives both a complete list of all the invariants and their explicit construction. (This is the most important sentence in this paper!)

We can therefore find all the invariants of type (p, q) by finding the direct sum decomposition of $1^{(p,0)}$ and $1^{(0,q)}$, taking their direct product, and identifying all the representations of weight 0 which occur in the direct sum decomposition of that direct product.

3.1 Some general remarks on invariant constraints

We note a few general properties of smoothness constraints of type (p, q) . Our first remark is that if the invariant

constraints of type $(p, 0)$ are $\{\mathcal{F}_i^{(p,0)}, i = 1, \dots, m\}$, and $\{\mathcal{F}_i^{(0,q)}, i = 1, \dots, n\}$ are the constraints of type $(0, q)$, then the mn quantities

$$\{\mathcal{F}_{i,j}^{(p,q)} \equiv \mathcal{F}_i^{(p,0)} \mathcal{F}_j^{(0,q)}, i = 1, \dots, m; j = 1, \dots, n\}$$

are all smoothness constraints of type (p, q) . We call such constraints *composite*.

Our second remark is that there is a relation between the constraints of type $(0, q)$, used for surface interpolation, and the *de-coupled* constraints of type $(q, 0)$ used for optical flow computations. If we show the explicit functional dependence of the former constraints as

$$\{\mathcal{F}_i^{(0,q)}[I], i = 1, \dots, n\},$$

Then it is easy to see that the decoupled constraints given by

$$\{\mathcal{F}_i[U] = \mathcal{F}_i^{(0,q)}[u] + \mathcal{F}_i^{(0,q)}[v]; i = 1, \dots, n\}.$$

are decoupled constraints of type $(q, 0)$. This is, for instance, the relation between the smoothness constraint used by Anandan and Weiss [Anan85] for computing optical flow, and the "quadratic variation" smoothness constraint introduced by Brady and Horn [Brad83] and by Grimson [Grim81].

In the next two sections, we find the direct sum decomposition of $1^{(p,0)}$ and $1^{(0,q)}$.

3.2 The direct sum decomposition of $1^{(p,0)}$ ($p = 1, 2$)

According to the remarks in the previous section, we have

$$\partial_a U_r \sim 1 \otimes 1,$$

and so

$$\partial_a U_r \partial_b U_s \sim (1 \otimes 1) \otimes (1 \otimes 1) = 1^{(1,0)}.$$

Our goal is to find the direct sum decomposition of this direct product.

It is shown in [Snyd90a] that

$$\partial_a U_r \sim 1 \otimes 1 = 0_1 \oplus 0_2 \oplus 2_2,$$

where, according to (9) and (10),

$$0_1 \sim \partial_1 U_1 + \partial_2 U_2 = u_x + v_y \equiv \lambda_1 \quad (20)$$

$$0_2 \sim \partial_2 U_1 - \partial_1 U_2 = u_y - v_x \equiv \lambda_2 \quad (21)$$

$$2_1 \sim \begin{pmatrix} \partial_1 U_1 - \partial_2 U_2 \\ \partial_2 U_1 + \partial_1 U_2 \end{pmatrix} = \begin{pmatrix} u_x - v_y \\ u_y + v_x \end{pmatrix} \equiv K^{(1)}. \quad (22)$$

Therefore

$$\partial_a U_r \partial_b U_s \sim (1 \otimes 1)(1 \otimes 1) = (0_1 \oplus 0_2 \oplus 2_1) \otimes (0_1 \oplus 0_2 \oplus 2_1).$$

We show in [Snyd90a] that this direct product has the direct sum decomposition:

$$(\partial U)^2 \sim 1^{(1,0)} \simeq 0_3 \oplus 0_4 \oplus 0_5 \oplus 0_6 \oplus 2_2 \oplus 2_3 \oplus 4_1, \quad (23)$$

where

$$0_3 = \lambda_1^2 \equiv \mathcal{F}_1^{(10)}$$

$$0_4 = \lambda_1 \lambda_2 \equiv \mathcal{F}_2^{(10)}$$

$$0_5 = \lambda_2^2 \equiv \mathcal{F}_3^{(10)}$$

$$0_6 = K^{(1)} \cdot K^{(1)} \equiv \mathcal{F}_4^{(10)}$$

$$2_2 = \lambda_1 K^{(1)} \quad (24)$$

$$2_3 = \lambda_2 K^{(1)} \quad (25)$$

$$4_1 \sim M^{(1)} \equiv \begin{pmatrix} (u_x - v_y)^2 - (u_y + v_x)^2 \\ 2(u_x - v_y)(u_y + v_x) \end{pmatrix}. \quad (26)$$

Here we use the symbol “ \simeq ” to mean that on the right-hand-side, all multiplicities have been set equal to one so that we do not have any redundant quantities (see [Snyd90a] for a discussion).

We note that in the process of finding the direct sum decomposition of $1^{(1,0)}$, we have also found that the invariant smoothness constraints of type (1,0) are given simply by the four invariants in (23), namely

$$\{\mathcal{F}_1^{(10)}, \mathcal{F}_2^{(10)}, \mathcal{F}_3^{(10)}, \mathcal{F}_4^{(10)}\}. \quad (27)$$

It is easy to check that these are linearly independent, and that the only de-coupled invariant smoothness constraint that can be formed by taking linear combinations of these is just the smoothness constraint \mathcal{F}_{H-S} of Horn and Schunck [Horn81]:

$$u_x^2 + u_y^2 + v_x^2 + v_y^2 = \mathcal{F}_{H-S} = \mathcal{F}_1^{(10)} + \mathcal{F}_3^{(10)} + \mathcal{F}_4^{(10)}.$$

We note that if the invariant of type (0,1) (given by equation (38) is denoted by $F[I]$, then the decoupled invariant of type (1,0) is just of the form

$$F[u] + F[v].$$

This is an example of the general comments made in Section 3.1.

A similar analysis for the direct sum decomposition of $1^{(2,0)}$ can then be made, whereupon [Snyd90a] we find (neglecting multiplicities):

$$(\partial^2 U)^2 \sim 1^{(2,0)} \simeq [0_7 \oplus 0_8 \oplus 0_9 \oplus 0_{10} \oplus 0_{11}] \oplus [2_4 \oplus 2_5 \oplus 2_6 \oplus 2_7 \oplus 2_8] \oplus [4_2 \oplus 4_3] \oplus 6_1, \quad (28)$$

where (with $\partial^+ \equiv \partial_{xx} - \partial_{yy}$ and $\partial^- \equiv 2\partial_x \partial_y$)

$$0_7 \sim \mathcal{F}_1^{(20)} \equiv (\nabla^2 u)^2 + (\nabla^2 v)^2$$

$$0_8 \sim \mathcal{F}_2^{(20)} \equiv (\partial^+ u + \partial^- v)^2 + (\partial^- u - \partial^+ v)^2$$

$$0_9 \sim \mathcal{F}_3^{(20)} \equiv \nabla^2 u (\partial^+ u + \partial^- v) + \nabla^2 v (\partial^- u - \partial^+ v)$$

$$0_{10} \sim \mathcal{F}_4^{(20)} \equiv \nabla^2 u (\partial^- u - \partial^+ v) - \nabla^2 v (\partial^+ u + \partial^- v)$$

$$0_{11} \sim \mathcal{F}_5^{(20)} \equiv (\partial^+ u - \partial^- v)^2 + (\partial^- u + \partial^+ v)^2$$

$$2_i \sim K_i \quad (i = 4, 5, 6, 7, 8)$$

$$4_j \sim M_j \quad (j = 2, 3),$$

$$6_1 \sim N_1 \equiv \begin{pmatrix} (\partial^+ u - \partial^- v)^2 - (\partial^- u + \partial^+ v)^2 \\ 2(\partial^+ u - \partial^- v)(\partial^- u + \partial^+ v) \end{pmatrix},$$

where

$$K^{(4)} \equiv \begin{pmatrix} (\nabla^2 u)^2 - (\nabla^2 v)^2 \\ 2\nabla^2 u \nabla^2 v \end{pmatrix} \quad (29)$$

$$K^{(5)} \equiv \begin{pmatrix} (\partial^+ u + \partial^- v)^2 - (\partial^- u - \partial^+ v)^2 \\ 2(\partial^+ u + \partial^- v)(\partial^- u - \partial^+ v) \end{pmatrix} \quad (30)$$

$$K^{(6)} \equiv \begin{pmatrix} \nabla^2 u (\partial^+ u + \partial^- v) - \nabla^2 v (\partial^- u - \partial^+ v) \\ \nabla^2 u (\partial^- u - \partial^+ v) + \nabla^2 v (\partial^+ u + \partial^- v) \end{pmatrix} \quad (31)$$

$$K^{(7)} \equiv \begin{pmatrix} \nabla^2 u (\partial^+ u - \partial^- v) + \nabla^2 v (\partial^- u + \partial^+ v) \\ \nabla^2 u (\partial^- u + \partial^+ v) - \nabla^2 v (\partial^+ u - \partial^- v) \end{pmatrix} \quad (32)$$

$$K^{(8)} \equiv \begin{pmatrix} (\partial^+ u + \partial^- v)(\partial^+ u - \partial^- v) \\ (\partial^+ u + \partial^- v)(\partial^- u + \partial^+ v) \end{pmatrix} + \begin{pmatrix} (\partial^- u - \partial^+ v)(\partial^- u + \partial^+ v) \\ (\partial^- u - \partial^+ v)(\partial^+ u - \partial^- v) \end{pmatrix} \quad (33)$$

$$M^{(2)} \equiv \begin{pmatrix} \nabla^2 u (\partial^+ u - \partial^- v) - \nabla^2 v (\partial^- u + \partial^+ v) \\ \nabla^2 u (\partial^- u + \partial^+ v) + \nabla^2 v (\partial^+ u - \partial^- v) \end{pmatrix} \quad (34)$$

$$M^{(3)} \equiv \begin{pmatrix} (\partial^+ u + \partial^- v)(\partial^+ u - \partial^- v) \\ (\partial^+ u + \partial^- v)(\partial^- u + \partial^+ v) \end{pmatrix} - \begin{pmatrix} (\partial^- u - \partial^+ v)(\partial^- u + \partial^+ v) \\ (\partial^- u - \partial^+ v)(\partial^+ u - \partial^- v) \end{pmatrix} \quad (35)$$

We note that there are here five invariants of type (2,0):

$$\{\mathcal{F}_1^{(20)}, \mathcal{F}_2^{(20)}, \mathcal{F}_3^{(20)}, \mathcal{F}_4^{(20)}, \mathcal{F}_5^{(20)}\}$$

where (expanding out the expressions above):

$$\mathcal{F}_1^{(20)} = (u_{xx} + u_{yy})^2 + (v_{xx} + v_{yy})^2$$

$$\mathcal{F}_2^{(20)} = (u_{xx} - u_{yy} + 2v_{xy})^2 + (v_{xx} - v_{yy} - 2u_{xy})^2$$

$$\mathcal{F}_3^{(20)} = u_{xx}^2 - u_{yy}^2 - v_{xx}^2 + v_{yy}^2 + 2u_{xy}(u_{xx} + v_{yy}) + 2v_{xy}(u_{xx} + u_{yy})$$

$$\mathcal{F}_4^{(20)} = 2[u_{xy}(u_{xx} + u_{yy}) - v_{xy}(v_{xx} + v_{yy}) + u_{yy}v_{yy} - u_{xx}v_{xx}]$$

$$\mathcal{F}_5^{(20)} = (u_{xx} - u_{yy} - 2v_{xy})^2 + (v_{xx} - v_{yy} + 2u_{xy})^2$$

It is easy to show that these five invariant constraints are linearly independent. It is also easy to show that precisely two linear combinations of these invariants are decoupled. $\mathcal{F}_1^{(20)} \equiv \mathcal{G}_1^{(20)}$ is manifestly so, and so is the linear combination

$$\mathcal{G}_2^{(20)} \equiv \frac{2\mathcal{F}_1^{(20)} + \mathcal{F}_2^{(20)} + \mathcal{F}_5^{(20)}}{4} = u_{xx}^2 + 2u_{xy}^2 + u_{yy}^2 + \{u \rightarrow v\}.$$

It is clear that both of these constraints are positive definite. We note that $\mathcal{G}_2^{(20)}$ is just the smoothness constraint used by Anandan and Weiss [Anan85] for computing optical flow. We note also that the decoupled constraints of type (2,0), $\mathcal{G}_1^{(20)}$ and $\mathcal{G}_2^{(20)}$, are related to those of type (0,2) (cf. the discussion following equation (37), as was remarked upon in Section 3.1.

In summary, then, there are 5 scalar smoothness constraints of type (2,0). The two decoupled ones are:

$$\begin{aligned}\mathcal{G}_1^{(20)} &= (u_{xx} + u_{yy})^2 + (v_{xx} + v_{yy})^2 \\ \mathcal{G}_2^{(20)} &= [(u_{xx} - v_{yy})^2 + 4u_{xy}^2] + [(v_{xx} - u_{yy})^2 + 4v_{xy}^2]\end{aligned}$$

and the three coupled ones can be taken to be:

$$\begin{aligned}\mathcal{F}_3^{(20)} &= u_{xx}^2 - u_{yy}^2 - v_{xx}^2 + v_{yy}^2 + \\ &\quad + 2u_{xy}(v_{xx} + v_{yy}) + 2v_{xy}(u_{xx} + u_{yy}) \\ \frac{1}{8}(\mathcal{F}_2^{(20)} - \mathcal{F}_5^{(20)}) &= (u_{xx} - u_{yy})v_{xy} - (v_{xx} - v_{yy})u_{xy} \\ \frac{1}{2}\mathcal{F}_2^{(20)} &= u_{xy}(u_{xx} + u_{yy}) - v_{xy}(v_{xx} + v_{yy}) + \\ &\quad + u_{yy}v_{yy} - u_{xx}v_{xx}.\end{aligned}$$

3.3 The Decomposition of $1^{(0,1)}$ and $1^{(0,2)}$

We will also need the direct sum decomposition of quantities quadratic in 1st and 2nd derivatives of the grey-level intensity function I . We find that

$$(\partial I)^2 \simeq 0_{12} \oplus 2_0 \quad (36)$$

$$(\partial^2 I)^2 \simeq 0_{13} \oplus 0_{14} \oplus 2_{10} \oplus 4_4, \quad (37)$$

where

$$0_{12} \sim |\nabla I|^2 \equiv \mathcal{F}_1^{(01)} \quad (38)$$

$$0_{13} \sim (\nabla^2 I)^2 \equiv \mathcal{F}_1^{(02)} \quad (39)$$

$$0_{14} \sim (I_{xx} - I_{yy})^2 + 4I_{xy}^2 \equiv \mathcal{F}_2^{(02)} \quad (40)$$

$$2_9 \sim K^{(9)} \equiv \begin{pmatrix} I_x^2 - I_y^2 \\ 2I_x I_y \end{pmatrix}. \quad (41)$$

$$2_{10} \sim K^{(10)} \equiv (\nabla^2 I) \begin{pmatrix} I_{xx} - I_{yy} \\ 2I_{xy} \end{pmatrix} \quad (42)$$

$$4_4 \sim M^{(4)} \equiv \begin{pmatrix} (I_{xx} - I_{yy})^2 - 4I_{xy}^2 \\ 4(I_{xx} - I_{yy})I_{xy} \end{pmatrix}. \quad (43)$$

We note that in finding these direct sum decompositions we have also found all the invariant smoothness constraints of types (0,1) and (0,2). Namely, the smoothness constraint of type (0,1) is just the single quantity $\mathcal{F}_1^{(01)}$ defined in (38), whereas the smoothness constraints of type (0,2) are two in number: $\mathcal{F}_1^{(02)}$ and $\mathcal{F}_2^{(02)}$, defined in (39) and (40), respectively. These constraints are manifestly positive definite, and hence are suitable for "performance functions" for surface interpolation. Our results therefore constitute a

generalization of the results of [Brad83] and [Grim81], who also gave completeness proofs for smoothness constraints of this type, but for the more restricted case of "rotational symmetry". Our method is, we believe, more direct than that of the aforementioned authors. We remark also that $\mathcal{F}_1^{(02)}$ has been called by Grimson [Grim81] the "squared Lagrangian," and that the (positive definite) linear combination

$$\frac{1}{2}[\mathcal{F}_1^{(02)} + \mathcal{F}_2^{(02)}] = I_{xx}^2 + 2I_{xy}^2 + I_{yy}^2 \equiv \mathcal{F}_+^{(02)}$$

has been called by him the "quadratic variation." As is well known, the quantity $\mathcal{F}_1^{(02)} - \mathcal{F}_+^{(02)}$ is a total derivative (the divergence of a vector).

4 Invariant Smoothness Constraints of Type (1,1) and (1,2)

Now that we have found the direct sum decompositions of the portion of each smoothness constraint of type (p,q) which depends only on the optical flow, or only on the image intensity, it is a simple matter to find the invariant constraints for arbitrary p and q . We limit ourselves in this section to the cases $(p,q) = (1,1), (1,2)$.

Before proceeding, we note that some of the invariants of type (p,q) can be written down immediately. Namely, the product of an invariant of type $(p,0)$ and one of type $(0,q)$ is clearly an invariant of type (p,q) . We shall call such invariants "composite invariants of type (p,q) ." Since there are 4 invariants of type (1,0), 2 of type (2,0), 1 of type (0,1), and 2 of type (0,2), it follows that there are $4 \times 1 = 4$ composite invariants of type (1,1), $4 \times 2 = 8$ of type (1,2), $2 \times 1 = 2$ of type (2,1), and $2 \times 2 = 4$ of type (2,2). We will see this explicitly in the remainder of this Section and in Section 5

4.1 Invariant Smoothness Constraints of Type (1,1)

These smoothness constraints are quadratic in 1st derivatives of the optical flow and of the image intensity function:

$$(\partial_a U, \partial_b U, \partial_c I \partial_d I \sim (\partial U)^2 (\partial I)^2 \quad (44)$$

From (23) and (36), we have that (44) transforms like the direct product representation

$$\partial_a U, \partial_b U, \partial_c I \partial_d I \simeq \left\{ \left(\sum_{i=1}^{i=6^0} 0_i \right) \oplus 2_2 \oplus 2_3 \oplus 4_1 \right\} \otimes (0_{12} \oplus 2_0).$$

Here, we have defined

$$\sum_{i=1}^{i=n^0} m_i \equiv m_1 \oplus \cdots \oplus m_n.$$

We are only interested in the terms in the direct sum decomposition that transform like 0. These can arise only from the terms of the form $m \otimes m$ in the above expression. Thus:

$$\partial_a U_r \partial_b U_s \partial_c I \partial_d I \sim \left(\sum_{i=3}^{i=6} (0_i \otimes 0_{12}) \right) \oplus (2_2 \otimes 2_9) \oplus (2_3 \otimes 2_9) \oplus \{O.T.\},$$

where {O.T.} represents the terms which cannot give rise to scalars. We then find the direct sum decomposition of each of these double products, and identify the resulting scalars. The result is the eight invariant smoothness constraints of type (1,1):

$$\begin{aligned} \mathcal{F}_1^{(11)} &= \mathcal{F}_1^{(01)} \mathcal{F}_1^{(10)} = |\nabla I|^2 (u_x + v_y)^2 \\ \mathcal{F}_2^{(11)} &= \mathcal{F}_1^{(01)} \mathcal{F}_2^{(10)} = |\nabla I|^2 (u_x + v_y)(u_y - v_x) \\ \mathcal{F}_3^{(11)} &= \mathcal{F}_1^{(01)} \mathcal{F}_3^{(10)} = |\nabla I|^2 (u_y - v_x)^2 \\ \mathcal{F}_4^{(11)} &= \mathcal{F}_1^{(01)} \mathcal{F}_4^{(10)} = |\nabla I|^2 (u_y + v_x)^2 + (u_x - v_y)^2 \\ \mathcal{F}_5^{(11)} &= (u_x + v_y)[(u_x - v_y)(I_x^2 - I_y^2) + (u_y + v_x)(2I_x I_y)] \\ \mathcal{F}_6^{(11)} &= (u_x + v_y)[(u_y + v_x)(I_x^2 - I_y^2) + (v_y - u_x)(2I_x I_y)] \\ \mathcal{F}_7^{(11)} &= (u_y - v_x)[(u_x - v_y)(I_x^2 - I_y^2) + (u_y + v_x)(2I_x I_y)] \\ \mathcal{F}_8^{(11)} &= (u_y - v_x)[(u_y + v_x)(I_x^2 - I_y^2) + (v_y - u_x)(2I_x I_y)] \end{aligned}$$

where the $\mathcal{F}_i^{(10)}$ are the invariant constraints of type (1,0).

It is easy to check that the eight invariants listed above are, in fact, linearly independent. One can further show that there are exactly three independent linear combinations of these eight invariants which are decoupled constraints (of type (1,1)):

$$\begin{aligned} \mathcal{G}_1^{(11)} &= \mathcal{F}_1^{(11)} + \mathcal{F}_3^{(11)} + \mathcal{F}_4^{(11)} \\ &= |\nabla I|^2 (u_x^2 + u_y^2 + v_x^2 + v_y^2) \equiv |\nabla I|^2 \mathcal{F}_{H-S} \\ \mathcal{G}_2^{(11)} &= \mathcal{F}_5^{(11)} - \mathcal{F}_8^{(11)} = (u_x^2 + v_x^2 - u_y^2 - v_y^2)(I_x^2 - I_y^2) + \\ &\quad (u_x u_y + v_x v_y)(4I_x I_y) \\ \mathcal{G}_3^{(11)} &= \mathcal{F}_6^{(11)} + \mathcal{F}_7^{(11)} = (u_x^2 + v_x^2 - u_y^2 - v_y^2)(-2I_x I_y) + \\ &\quad (u_x u_y + v_x v_y)(2(I_x^2 - I_y^2)). \end{aligned}$$

These are related to the two decoupled constraints of type (1,1) defined in [Snyd89] (but due to Nagel and Enkelmann [Nage86] (see also [Nage83, Nage87])). See [Snyd90a] for the details.

4.2 ISO(2) invariant smoothness constraints of type (1,2)

In this section, we consider smoothness constraints of type (1,2), i.e., combinations of the form

$$(\partial_a U_r \partial_b U_s) \partial_c \partial_d I \partial_m \partial_n I \quad (45)$$

which are invariant. From (23) and (37) we have that a quantity of the form (45) will transform under ISO(2) like the direct product

$$(\partial U)^2 (\partial^2 I)^2 \simeq \left(\sum_{i=3}^{i=6} 0_i \right) \oplus 2_2 \oplus 2_3 \oplus 4_1 \otimes (0_{13} \oplus 0_{14} \oplus 2_{10} \oplus 4_4).$$

We see immediately that there are $4 \times 2 = 8$ "composite" invariant $2 \times 2 = 4$ invariants coming from the two 0's in each of the two $2 \otimes 2$'s, and $2 \times 1 = 2$ invariants coming from the two 0's in the $4_1 \otimes 4_4$, making a total of $8 + 4 + 2 = 14$ invariants of type (1,2). These can be easily constructed to yield:

$$\begin{aligned} \mathcal{F}_1^{(12)} &\equiv \mathcal{F}_1^{(10)} \mathcal{F}_1^{(02)} = (u_x + v_y)^2 (I_{xx} + I_{yy})^2 \\ \mathcal{F}_2^{(12)} &\equiv \mathcal{F}_1^{(10)} \mathcal{F}_2^{(02)} = (u_x + v_y)^2 [(I_{xx} - I_{yy})^2 + 4I_{xy}^2] \\ \mathcal{F}_3^{(12)} &\equiv \mathcal{F}_2^{(10)} \mathcal{F}_1^{(02)} = (u_x + v_y)(u_y - v_x)(I_{xx} + I_{yy})^2 \\ \mathcal{F}_4^{(12)} &\equiv \mathcal{F}_2^{(10)} \mathcal{F}_2^{(02)} = (u_x + v_y)(u_y - v_x)[(I_{xx} - I_{yy})^2 + 4I_{xy}^2] \\ \mathcal{F}_5^{(12)} &\equiv \mathcal{F}_3^{(10)} \mathcal{F}_1^{(02)} = (u_y - v_x)^2 (I_{xx} + I_{yy})^2 \\ \mathcal{F}_6^{(12)} &\equiv \mathcal{F}_3^{(10)} \mathcal{F}_2^{(02)} = (u_y - v_x)^2 [(I_{xx} - I_{yy})^2 + 4I_{xy}^2] \\ \mathcal{F}_7^{(12)} &\equiv \mathcal{F}_4^{(10)} \mathcal{F}_1^{(02)} = [(u_x - v_y)^2 + (u_y + v_x)^2](I_{xx} + I_{yy})^2 \\ \mathcal{F}_8^{(12)} &\equiv \mathcal{F}_4^{(10)} \mathcal{F}_2^{(02)} = [(u_x - v_y)^2 + (u_y + v_x)^2][(I_{xx} - I_{yy})^2 + 4I_{xy}^2] \\ \mathcal{F}_9^{(12)} &\equiv (u_x + v_y)[(u_x - v_y)(I_{xx}^2 - I_{yy}^2) + (u_y + v_x)(2I_{xy} \nabla^2 I)] \\ \mathcal{F}_{10}^{(12)} &\equiv (u_x + v_y)[(u_y + v_x)(I_{xx}^2 - I_{yy}^2) - (u_x - v_y)(2I_{xy} \nabla^2 I)] \\ \mathcal{F}_{11}^{(12)} &\equiv (u_y - v_x)[(u_x - v_y)(I_{xx}^2 - I_{yy}^2) + (u_y + v_x)(2I_{xy} \nabla^2 I)] \\ \mathcal{F}_{12}^{(12)} &\equiv (u_y - v_x)[(u_y + v_x)(I_{xx}^2 - I_{yy}^2) - (u_x - v_y)(2I_{xy} \nabla^2 I)] \\ \mathcal{F}_{13}^{(12)} &\equiv ((u_x - v_y)^2 - (u_y + v_x)^2)((I_{xx} - I_{yy})^2 - 4I_{xy}^2) + \\ &\quad + (2(u_x - v_y)(u_y + v_x))(4I_{xy}(I_{xx} - I_{yy})) \\ \mathcal{F}_{14}^{(12)} &\equiv ((I_{xx} - I_{yy})^2 - 4I_{xy}^2)(2(u_x - v_y)(u_y + v_x)) - \\ &\quad - (4I_{xy}(I_{xx} - I_{yy}))((u_x - v_y)^2 - (u_y + v_x)^2). \end{aligned}$$

One can show that all fourteen of these are linearly independent. Further, one can show that there are exactly four linear combinations of the $\mathcal{F}_i^{(12)}$ which are decoupled:

$$\begin{aligned} \mathcal{G}_1^{(12)} &\equiv \frac{1}{2} [\mathcal{F}_1^{(12)} + \mathcal{F}_5^{(12)} + \mathcal{F}_7^{(12)}] = \mathcal{F}_{H-S} [\nabla^2 I]^2 \\ \mathcal{G}_2^{(12)} &\equiv \frac{1}{2} [\mathcal{F}_2^{(12)} + \mathcal{F}_6^{(12)} + \mathcal{F}_8^{(12)}] = \mathcal{F}_{H-S} [(I_{xx} - I_{yy})^2 + 4I_{xy}^2] \\ \mathcal{G}_3^{(12)} &\equiv \frac{1}{2} [\mathcal{F}_{10}^{(12)} + \mathcal{F}_{11}^{(12)}] = (I_{xx}^2 - I_{yy}^2)(u_x u_y + v_x v_y) + \\ &\quad + 2I_{xy} \nabla^2 I (-u_x^2 + u_y^2 - v_x^2 + v_y^2) \\ \mathcal{G}_4^{(12)} &\equiv \frac{1}{2} [\mathcal{F}_9^{(12)} - \mathcal{F}_{12}^{(12)}] = (I_{xx}^2 - I_{yy}^2)(u_x^2 - u_y^2 + v_x^2 - v_y^2) + \\ &\quad + 2I_{xy} \nabla^2 I (u_x u_y + v_x v_y) \end{aligned}$$

The relation between these decoupled invariants and the two positive definite ones found in [Snyd89] (originally

proposed by Nagel and Enkelmann [Nage86]) is given in [Snyd90a].

5 Invariants of Type (2,1) and (2,2)

As far as we are aware, smoothness constraints quadratic in the 2nd derivative of optical flow have been considered only by Anandan and Weiss [Anan85], who used a decoupled constraint of type (2,0). Although there are probably good reasons why such constraints have been neglected (they would probably be sensitive to small errors), this may be an artifact of present algorithms. At any rate, in [Snyd90a] we show that there are 15 invariants of type (2,1), two of which are decoupled, and that there are 24 invariants of type (2,2), six of which are decoupled. The interest in constraints of the latter type is that unlike the other cases, there are two decoupled constraints which are *not* products of lower-ranking invariants. These are given by:

$$G_1^{22} = (I_{xx}^2 - I_{yy}^2)(u_{xx}^2 - u_{yy}^2) + 4I_{xy}\nabla^2 I[u_{xy}\nabla^2 u] + \{u \rightarrow v\}$$

$$G_2^{22} = (I_{xx}^2 - I_{yy}^2)[u_{xy}\nabla^2 u]I_{xy}\nabla^2 I[u_{xx}^2 - u_{yy}^2] + \{u \rightarrow v\}$$

6 Summary and Conclusions

We have used the representation theory of ISO(2) to give a complete list of all possible ISO(2)-invariant smoothness constraints which are quadratic in p^{th} derivatives of the optical flow, and in q^{th} derivatives of the grey-level intensity, for $p, q \leq 2$. We also found all the possible smoothness constraints for surface interpolation which were quadratic in 1st or 2nd. In [Snyd90a] this is extended to constraints of type (0,3) and (0,4). In the sequel to this work [Snyd90b], we use the complex formulation of the representation theory of SO(2) to find explicit formulas for the performance function quadratic in n^{th} derivatives, for arbitrary n . We find that the number of such invariants is given by $\{n/2\} + 1$ (where $\{x\}$ is the integer part of x), as well as their explicit form.

Acknowledgements

This research has been supported by the Defense Advanced Research Projects Agency under RADC contract F30602-87-C-0140 and Army ETL contract DACA76-89-C-0017.

References

- [Anan85] P. Anandan and R. Weiss, "Introducing a Smoothness Constraint in a Matching Approach for the Computation of Displacement Fields," UMass COINS Tech. Rep. 85-38 (Dec. 1985).
- [Brad83] M. Brady and B. K. P. Horn, "Rotationally Symmetric Operators for Surface Interpolation," CVGIP **22**, 70-94 (1983).
- [Grim81] W. E. L. Grimson, *From Images to Surfaces: A Computational Study of the Human Early Visual System*, MIT Press, Cambridge, Mass. (1981).
- [Horn81] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow," Art. Intell. **23**, 175-203 (1981).
- [Murn38] F. D. Murnaghan, *The Theory of Group Representations*, Dover, NY, NY (1938).
- [Nage83] H.-H. Nagel, "Constraints for the Estimation of Displacement Vector Fields from Image Sequences," Proc. IJCAI83, 945-951, Karlsruhe, W. Germany (1983).
- [Nage86] H.-H. Nagel and W. Enkelmann, "An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fields from Image Sequences," IEEE Trans. Patt. An. Mach. Intell. PAMI-8, No. 5, 565-593 (Sept. 1986).
- [Nage87] H.-H. Nagel, "On the Estimation of Optical Flow: Relations between Different Approaches and Some New Results," Art. Intell. **33**, 299-324 (1987).
- [Snyd89] M. A. Snyder, "On the Mathematical Foundations of Smoothness Constraints for the Determination of Optical Flow and for Surface Reconstruction," COINS Tech. Rep. 89-05, Univ. of Mass., Amherst (January 1989); also in Proc. of Workshop on Motion, Irvine, Calif. (March 1989).
- [Snyd90a] M. A. Snyder, "The Mathematical Foundations Of Smoothness Constraints: The Case of Coupled Constraints," COINS Tech. Rep. in preparation.
- [Snyd90b] M. A. Snyder, "Closed-form Solutions for Smoothness Constraints of Type (p,q) for Arbitrary p and q," COINS Tech. Rep. in preparation.

A PHOTOMETRIC INVARIANT AND SHAPE CONSTRAINTS AT PARABOLIC POINTS

Lawrence B. Wolff¹
Computer Science Department
Columbia University
New York, N.Y. 10027

ABSTRACT

Recent research has revealed the invariance of the direction of image curves of equal reflected radiance (i.e., isophotes) from smooth surfaces at parabolic points with respect to all possible smooth reflectance maps assuming a fixed vantage point. This has enabled the relatively easy detection of parabolic points in images from shading analysis. For smooth surfaces possessing curves or 2-D regions consisting of parabolic points, the identification of these curves and regions provides important information about the structure of the surface. This paper enriches this photometric invariant for isophotes by using differential geometry techniques to show that the invariant direction of isophotes in the image plane for pixels corresponding to parabolic points has significant geometric meaning; namely it is the projection of the principal direction with curvature zero. Therefore we demonstrate that this photometric invariant at parabolic points is directly related to the geometry of the surface at such points. Furthermore, we carefully discuss under which physical imaging conditions this photometric invariance property breaks down and when it is best approximated, generalizing from orthographic images to perspective images. We present experimental evidence of this photometric invariance and discuss applications to constraints on surface orientation and curvature at parabolic points.

Topic Area: 3-D vision, shape from shading, differential geometry

1 INTRODUCTION

Parabolic points on smooth surfaces are points which possess one principal curvature of value zero. Planar points are trivially parabolic, but the subject matter discussed in this paper is not applicable to these points due to lack of shading information. A *parabolic line* on a smooth surface is a curve whose locus of points are all parabolic. As pointed out in [Koenderink and van Doorn 1980], a parabolic line on a smooth surface cleanly divides *elliptic* and *hyperbolic* surface regions. Elliptic points have principal curvatures which are both positive, and hyperbolic points have principal curvatures with opposite sign. The identification of parabolic lines on a smooth surface therefore gives a structural cue designating the boundary between "saddle-like" hyperbolic points and points which are completely concave or convex. Two dimensional neighborhoods of parabolic points on a smooth surface are neighborhoods which resemble exactly the surface structure of certain ruled surfaces including the cylinder and the cone. Figure 1 shows some examples of parabolic lines on various surfaces.

¹This work was supported in part by ARPA grant #N00039-84-C-0165 and NSF grant IRI-88-00370. This work was supported in part by an IBM Graduate Fellowship Award.

Interest in shading analysis at parabolic points on smooth surfaces began with the paper by [Koenderink and van Doorn 1980]. They showed that the local direction of lines of equal reflected radiance (i.e., isophotes) at parabolic points on Lambertian surfaces are invariant to varying illumination conditions, viewed from a fixed vantage point. Later the paper by [Blake, Zisserman and Knowles 1985] suggests a clever way of showing that this photometric invariance property of the direction of isophotes at parabolic points holds with respect to all possible smooth reflectance maps (i.e., continuously differentiable reflectance functions) as functions of surface orientation. From the image irradiance equation

$$I(x, y) = R(p, q)$$

they considered the resulting matrix equation in image coordinates (x, y) and in gradient coordinates (p, q) , from [Woodham 1978]:

$$\begin{pmatrix} I_x \\ I_y \end{pmatrix} = \begin{pmatrix} \partial p / \partial x & \partial q / \partial x \\ \partial p / \partial y & \partial q / \partial y \end{pmatrix} \begin{pmatrix} R_p \\ R_q \end{pmatrix} \quad (1)$$

where subscripting means partial differentiation. The 2x2 matrix of partial derivatives is the transpose of the Jacobian transformation between image coordinates and gradient coordinates. The

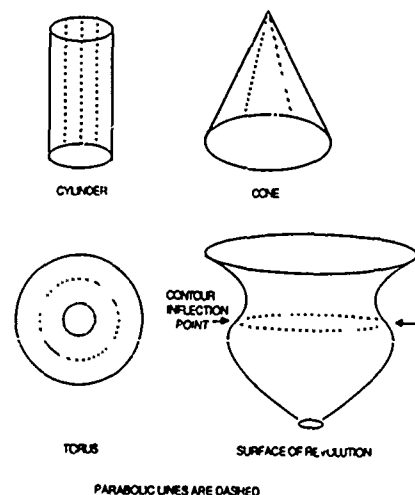


Figure 1:

Jacobian transformation between image coordinates and gradient coordinates is the *viewer centered curvature matrix* relating the rate of change of the surface normal, expressed in gradient coordinates, to a corresponding displacement vector in the image coordinate system. When an image coordinate displacement vector is multiplied by the viewer centered curvature matrix, the resulting vector is the rate of change of the surface normal along this image direction, expressed in gradient coordinates. In the approximation of local orthographic imaging (i.e., rays of projection change very little between adjacent pixels) the 2x2 matrix of partial derivatives in equation 1 becomes symmetric as the object surface can be locally parameterized as a (possibly foreshortened) height function over image coordinates. Under such conditions this 2x2 matrix becomes the viewer centered curvature matrix.

[Blake Zisserman and Knowles 1985] point out that the diagonalization of the viewer centered curvature matrix at parabolic points is a scalar multiple of:

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

in some transformed image coordinate system. Since the image intensity gradient is related to the reflectance function gradient according to equation 1, the image intensity gradient will be parallel to the eigenvector direction, with non-zero eigenvalue, in the image plane for all smooth reflectance maps.

In section 2, we present a more formal derivation of the directional invariance of the image intensity gradient at parabolic points for smooth reflectance maps as a function of surface orientation. In section 3 we generalize to smooth reflectance maps as a function of an arbitrary number of physical parameters. This formality allows us to carefully analyze the physical imaging conditions under which the directional invariance of the image intensity gradient at parabolic points breaks down. Even though such a photometric invariance theoretical breaks down under some common circumstances, we show under which practical conditions it is at least well approximated. We also consider edges of specularities to be isophotes, and therefore apply the same directional invariance to these edges passing through parabolic points.

In section 4 we show that image isophotes at parabolic points are parallel to the projection of the principal direction of zero curvature, under conditions of local orthographic imaging. We depend on the theoretical development in section 2. The basic idea is that the tangent direction to isophote curves at parabolic points are shown to be a zero of the viewer centered curvature matrix. That is, the viewer centered curvature matrix multiplied by this tangent direction is the zero vector. We prove a theorem stating that under orthographic imaging, a direction in the image plane is a zero of the viewer centered curvature matrix if and only if it is the projection of the principal direction of zero curvature. This involves relating the measure of surface curvature provided by the viewer centered curvature matrix to the measure of surface curvature provided by the differential of the Gauss map (which we term in this paper the *surface centered curvature matrix*). This theorem easily extends to local orthographic imaging.

In section 5 we show experimental evidence that the direction of isophotes at parabolic points provides a good cue for the projected principal direction of zero curvature. We then show how surface orientation can be constrained at parabolic points with the help of this cue. On straight cylinders with arbitrary smooth cross section (e.g., a cylinder with cross section being circular,

elliptical, clover-leafed, etc...) , all points are parabolic. Cross sections of these surfaces are principal lines of non-zero curvature. Isophotes give cues for the principal direction of zero curvature, and at intersections of isophotes with cross sections surface orientation is constrained by observing how the orthogonality of principal directions of curvature is skewed. In addition a "net" of rulings is provided by the directionality of isophotes along these cross section lines, which is useful to algorithms trying to recover the orientation of the axis of cylinders with arbitrary cross section [Ulupinar and Nevatia 1988]. This is only one basic application of this photometric invariant for shape determination. Hopefully other possible applications will arise in the future.

There has been much research done on determining shape from reflectance maps in monocular views. Shape from shading techniques [Horn 1970, Pentland 1984, Cernuschi-Frias and Cooper 1984, Lee and Rosenfeld 1985] utilize shading of diffuse reflectance. Shape from specularly [Thrift and Lee 1983, Babu Lee and Rosenfeld 1985, Healey and Binford 1987, Buchanan 1987, Brelstaff 1989] utilize the nature of specular reflection. All these methods depend upon some assumption about a reflectance map and/or imaging geometry. Even though the geometric implications of the photometric invariant presented in this paper is applicable only to parabolic points, the beauty is that whatever information is learned from such a photometric cue, is obtained essentially for free, assuming one knows where parabolic points are to start with. This paper does not concern itself with having to find parabolic points, and assumes that these are given. There are already some existing methods that do consider the problem of finding parabolic points [Koenderink and van Doorn 1980, Blake Zisserman and Knowles 1985, Ponce Chelberg and Mann 1988]. Perhaps this paper helps motivate the importance of developing even better methods to discern parabolic points.

2 DERIVATION OF THE IMAGE INTENSITY GRADIENT AT PARABOLIC POINTS

We derive an image coordinate representation of the image intensity gradient at parabolic points which directly demonstrates its directional invariance with respect to all smooth reflectance maps, $R(p, q)$, as a function of surface orientation gradient coordinates p and q . In the next section we generalize to reflectance maps as functions of an arbitrary set of physical parameters.

Consider a non-singular linear transformation M of image coordinates:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = M \begin{pmatrix} x \\ y \end{pmatrix}.$$

The resulting transformation on the gradient of the image intensity function, I , is as follows²:

$$M^\dagger \begin{pmatrix} I_{x'} \\ I_{y'} \end{pmatrix} = \begin{pmatrix} I_x \\ I_y \end{pmatrix}$$

where \dagger implies matrix transpose, and as usual subscripting im-

²The duality of the transformations on image coordinates and image gradient vectors with respect to coordinate transformations is common in tensor analysis. In this case, image coordinates are said to transform *contravariantly* while image intensity gradients are said to transform *covariantly*.

plies partial differentiation. Using equation 1 we get:

$$\begin{pmatrix} I_{x'} \\ I_{y'} \end{pmatrix} = M^{\dagger^{-1}} \begin{pmatrix} I_x \\ I_y \end{pmatrix} = M^{\dagger^{-1}} \begin{pmatrix} \partial p / \partial x & \partial q / \partial x \\ \partial p / \partial y & \partial q / \partial y \end{pmatrix} \begin{pmatrix} R_p \\ R_q \end{pmatrix} \\ = M^{\dagger^{-1}} \begin{pmatrix} \partial p / \partial x & \partial q / \partial x \\ \partial p / \partial y & \partial q / \partial y \end{pmatrix} M^{\dagger} M^{\dagger^{-1}} \begin{pmatrix} R_p \\ R_q \end{pmatrix}.$$

Assuming local orthographic projection the above matrix of partial derivatives is symmetric and therefore equivalent to the viewer centered curvature matrix. At a parabolic point there exists an image coordinate displacement vector along which the rate of change of the surface normal is zero. The viewer centered curvature matrix must therefore have a zero eigenvalue. Hence we can select a non-singular image coordinate transformation, M , such that a similarity transformation with respect to $M^{\dagger^{-1}}$ of the viewer centered curvature matrix produces:

$$\begin{pmatrix} I_{x'} \\ I_{y'} \end{pmatrix} = \begin{pmatrix} \lambda & 0 \\ 0 & 0 \end{pmatrix} M^{\dagger^{-1}} \begin{pmatrix} R_p \\ R_q \end{pmatrix} = \begin{pmatrix} \lambda(m_{11}R_p + m_{12}R_q) \\ 0 \end{pmatrix} \quad (2)$$

where

$$M^{\dagger^{-1}} = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix}.$$

Since we are diagonalizing a symmetric matrix (i.e., the viewer centered curvature matrix), $M^{\dagger^{-1}}$, is in fact a 2-D rotation matrix.

Clearly, for all smooth reflectance maps, $R(p, q)$, the image intensity gradient is always parallel to the x' image coordinate axis.

3 PHYSICAL CONDITIONS AND PRACTICAL CONSIDERATIONS

We detail in this section the ideal physical conditions under which the photometric invariant at parabolic points discussed in this paper holds. We also discuss what practical considerations are most important for the photometric invariant to be best approximated.

We can assume that the reflectance function, R , is a function of an arbitrary number of physical parameters. Suppose then that

$$I(x, y) = R(p, q, \rho_1, \rho_2, \dots, \rho_i, \dots)$$

where ρ_i can be any physical parameter (e.g., surface albedo, surface roughness, incident light source orientation, image coordinates, etc...). Then

$$\begin{pmatrix} I_{x'} \\ I_{y'} \end{pmatrix} = \begin{pmatrix} \partial p / \partial x & \partial q / \partial x \\ \partial p / \partial y & \partial q / \partial y \end{pmatrix} \begin{pmatrix} R_p \\ R_q \end{pmatrix} + \begin{pmatrix} \sum_i \frac{\partial \rho_i}{\partial x} R_{\rho_i} \\ \sum_i \frac{\partial \rho_i}{\partial y} R_{\rho_i} \end{pmatrix}. \quad (3)$$

To be general, we do not necessarily assume local orthographic imaging so that the matrix of partial derivatives in equation 3 may be the non-symmetric transpose of the viewer centered curvature matrix. However a 2x2 matrix and its transpose have the same characteristic equation [Hildebrand 1965] so that at a parabolic point the transpose of the viewer centered curvature matrix has a zero eigenvalue. Therefore we can diagonalize the 2x2 matrix of partial derivatives in equation 3 at a parabolic point as follows:

$$\begin{pmatrix} I_{x'} \\ I_{y'} \end{pmatrix} = \begin{pmatrix} \lambda & 0 \\ 0 & 0 \end{pmatrix} M^{\dagger^{-1}} \begin{pmatrix} R_p \\ R_q \end{pmatrix} + \begin{pmatrix} \sum_i \frac{\partial \rho_i}{\partial x} R_{\rho_i} \\ \sum_i \frac{\partial \rho_i}{\partial y} R_{\rho_i} \end{pmatrix}. \quad (4)$$

Comparing equation 4 with equation 2, the direction of the image intensity gradient will be invariant along the x' axis for all smooth reflectance maps, R , unless perturbed by a non-zero second component of the right most term in equation 4. Hence the directional invariance of the image intensity gradient is true if and only if the $\partial \rho_i / \partial y'$ are zero for each ρ_i .

For most practical purposes we would hope that there would be a number of parabolic points on an imaged object, in fact we would hope to find parabolic lines or 2-D parabolic regions. The transformation of image coordinates to diagonalize the 2x2 matrix of partial derivatives at each parabolic point will in general be different. Therefore the minimal necessity for the invariance of the direction of the image intensity gradient to hold for EVERY parabolic point is that the $\partial \rho_i / \partial y'$ are zero for each y' axis respective of each parabolic point. For simplicity, we can assume a slightly stronger (but less physically contrived) constraint on the physical parameters, ρ_i , that simply $\partial \rho_i / \partial x$ and $\partial \rho_i / \partial y$ are zero for each ρ_i . That is, image gradient directionality is invariant at parabolic points if the right most term of equation 3 is the zero vector.

For perspective images the precise invariance of the image intensity gradient at parabolic points is violated because in such a case the reflectance map, R , is necessarily dependent upon image coordinates x and y . For instance, given a flat surface in a perspective view the relative orientation of this surface varies from pixel to pixel as rays of projection onto these pixels are nonparallel. The constant orientation (p_0, q_0) exists with respect to one ray of projection (e.g., with respect to the optic axis). Hence the reflectance function will vary as the relative surface orientation varies from pixel to pixel. Also, according to [Horn and Sjöberg 1979] for pure pinhole perspective projection,

$$I \propto (\cos \alpha)^4 R$$

where α is the angle between the ray of projection and the optic axis. However this second mal-effect of perspective projection can be avoided by appropriately correcting image irradiance values according to which pixel they correspond too.

From our analysis here the invariance of the direction of isophotes at parabolic points holds ideally under physical conditions of orthographic projection, and no varying surface parameters, such as albedo or roughness, in the local vicinity of the parabolic point. Also, lighting elements must ideally be at a large distance from the object with respect to the size of the object in the field of the image. Otherwise incident light source orientation can vary across the image plane. This means that the direction of isophotes at parabolic points can be sensitive to mutual illumination effects.

The invariance of the image intensity gradient at parabolic points is closely approximated for practical purposes in most perspective images with cameras of sufficient pixel resolution. With a small relative variation in the orientation of rays of projection between adjacent pixels, the image intensity gradient at parabolic points is invariant up to a good approximation. With all other physical parameters fixed across the image plane except possibly for surface orientation p, q accounted for in the 2x2 matrix term,

consider the right most term of equation 4 for a perspective view:

$$\begin{pmatrix} R_{x'} \\ R_{y'} \end{pmatrix} = \begin{pmatrix} \partial p^{rop}/\partial x'R_p + \partial q^{rop}/\partial x'R_q \\ \partial p^{rop}/\partial y'R_p + \partial q^{rop}/\partial y'R_q \end{pmatrix} \quad (5)$$

where as usual subscripting implies partial differentiation. The superscript *rop* for *p* and *q* refers to orientation of the corresponding ray of projection, so the respective partial derivatives quantify how the orientations of the rays of projection locally vary from pixel to pixel in a perspective view (with respect to $x' - y'$ coordinates). Taking note of equations 2 and 4, the smaller in magnitude $\partial p^{rop}/\partial y'$ is with respect to λm_{11} , and the smaller in magnitude $\partial q^{rop}/\partial y'$ is with respect to λm_{12} , the smaller the perturbation of the direction of the image intensity gradient for different reflectance maps. This quantifies how close physical imaging conditions should approximate local orthographic imaging in order for the direction of image gradients at parabolic points to be useful. To give an idea about the size of the partial derivatives in a fairly wide perspective view, consider the rays of projection for a 30 degree half-cone angle field of view taken with a 512x512 pixel resolution camera. The size of the partial derivatives are approximately 1/443, which is very small. With all other physical parameters being constant at a parabolic point, equation 4 and equation 2 become identical in the limit as the partial derivatives of the orientation of rays of projection go to zero towards local orthographic projection.

The same analysis applies to distant light source approximation where the partial derivatives of surface orientation in equation 5 now refer to the rate of change of incident orientation of the light source between adjacent pixels. In both cases of local orthographic approximation and distant light source approximation, it is clear that the higher the non-zero viewer centered curvature eigenvalue, λ , the better the invariance of the direction of the isophote at the corresponding parabolic point.

The photometric invariant at parabolic points can be practically applied using edges of specularities that pass through these points. Within a specularity, intensity variations can be quite large in magnitude due to small surface orientation changes and non-uniformities in the light source. Isolating isophotes within a specularity is not practical. However specularities have well defined edges at where diffuse reflecting points usually exponentially increase in intensity (e.g., the Torrance-Sparrow reflectance model [Torrance and Sparrow 1967]) into the specularity region. Under the assumption that the specularity can be completely segmented using a single intensity threshold (i.e., the image intensities composing the specularity are all above the surrounding diffuse reflecting intensities), the image curve running along the edge of the specularity where it begins to dominate over the diffuse component approximates very closely the isophote at that constant intensity threshold. Hence the direction of specular edges at parabolic points, under the ideal physical conditions stated above (i.e., orthographic image projection, and distant light source), is invariant to how the object is situated with respect to the light source.

The biggest perturbation to a diffuse image intensity gradient at parabolic points is albedo and material variations that might occur at these points. Unless these variations are minor, the application of the photometric invariant discussed in this paper is not practical for diffuse reflection. However, in the case of large reflected diffuse intensity changes due to object albedo and/or material changes, the direction of imaged edges of specularities

passing through parabolic points will still be invariant, as long as the intensities within the specularity are large relative to the albedo/material diffuse intensity changes.

4 ISOPHOTES AT PARABOLIC POINTS

4.1 A THEOREM ABOUT CURVATURE MATRICES

Our main motivation for the mathematical development in this subsection is to prove a theorem which is essential to establishing the relationship between the direction of isophotes in the image plane at parabolic points and intrinsic surface geometry. We use the same style of notation as in [do Carmo 1976]. When we refer to a surface, S , the notation $S(x, y)$ implies a particular parameterization of that surface with respect to two dimensional coordinates x and y . Referring to a surface, $S(x, y)$, which is parameterized by height over the coordinates, (x, y) , implies there exists a twice differentiable height function $f(x, y)$ such that:

$$S(x, y) = (x, y, f(x, y)).$$

Note for the height parameterization, at occluding points where the surface normal is orthogonal to projection onto the image plane, the tangent plane appears collapsed to a line and does not exist for such a parameterization. Therefore our discussion excludes occluding points.

The theorem that is proved is as follows:

Theorem 1 Consider a surface, $S(x, y)$, parameterized by height over the image coordinates (x, y) . At a point on the surface, S , a direction in the tangent plane at this point is a principal direction of principal curvature zero if and only if the orthographic projection of this direction onto the image plane when multiplied by the viewer centered curvature matrix produces the zero vector.

This theorem establishes an equivalence between two measures of curvature of a smooth surface for a special direction at a parabolic point, namely the principal direction possessing zero curvature. We have encountered in the introduction the viewer centered curvature matrix which measures the rate of change of the surface normal at a point with respect to an image coordinate direction. Principal directions at a point on a surface and their respective curvatures are determined from another type of curvature matrix which measures how the surface normal is changing with respect to a local coordinate system defined on the tangent plane at the surface point. We will call this the surface centered curvature matrix. Looking at figure 2, the rate of change of the surface normal at a point with respect to a displacement by a vector v_{tp} in the tangent plane coordinate system is given by the surface centered curvature matrix multiplied by this vector. This rate of change of the surface normal is expressed in tangent plane coordinates. The above theorem states that the measure of the change of the surface normal by the surface centered curvature matrix and the viewer centered curvature matrix are equivalently zero for tangent plane and image plane displacements corresponding to the principal direction of zero curvature.

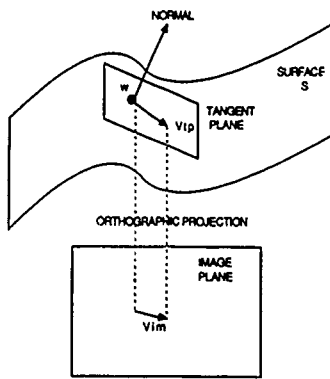


Figure 2:

What is termed here as the surface centered curvature matrix is more standardly known as the *differential of the Gauss map* (see [do Carmo 1976] pp.136-140 or [Besl and Jain 1985]). The *principal directions* are defined as the eigenvectors in the tangent plane of the surface centered curvature matrix and the *principal curvatures* are the respective eigenvalues. Therefore, by definition, the principal direction with curvature zero is the tangent plane direction for which the surface centered curvature matrix multiplies to zero.

Observing figure 2, with respect to moving along the surface S from point w , a displacement by the vector v_{tp} in the tangent plane is equivalent in the parameterized sense to a displacement by the vector v_{im} in the image plane. That is, S is parameterized by height and v_{im} is the orthographic projection of v_{tp} . The corresponding rate of change of the surface normal is expressed as a vector in the tangent plane at w by multiplying the surface centered curvature matrix by v_{tp} . The same rate of change of the surface normal is expressed as a vector in gradient coordinates by multiplying the viewer centered curvature matrix by v_{im} . Clearly if the rate of change of the surface normal is zero, this rate of change must be represented simultaneously by the zero vector in both tangent plane and gradient coordinates.

A more formal quantitative proof of this theorem is given in the appendix. It involves a bit more background in differential geometry, but the proof in the appendix shows the precise quantitative relationship between the surface centered and viewer centered curvature matrices, as well as the relationship between a local tangent plane coordinate system and gradient coordinates. These are useful geometric relationships in their own right.

Theorem 1 easily extends to a surface, S , which is parameterized by the image plane with respect to rays of projection at some constant angle between 0° and 90° to the image plane (i.e., local orthographic projection). Just consider the vector v_{im} to be the projection of v_{tp} along these rays.

4.2 THE DIRECTION OF ISOPHOTES AT PARABOLIC POINTS

We now prove the theorem that is central to this paper:

Theorem 2 *Assuming local orthographic projection, the tangent direction of an image isophote curve at a parabolic point is always parallel to the projection of the principal direction of zero curvature. This assumes that except for surface orientation, no other physical parameter varies at a parabolic point.*

The tangent direction of an image isophote curve is perpendicular to the image gradient. Since the theorem assumes local orthographic imaging, the viewer centered curvature matrix is symmetric and therefore has orthogonal eigenvectors. Looking at equation 2, the tangent direction of an isophote curve is therefore parallel to the y' axis which has eigenvalue zero as:

$$\begin{pmatrix} \lambda & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Hence from theorem 1, this direction must be the projection of the principal direction of zero curvature.

5 EXPERIMENTATION AND APPLICATIONS

5.1 EXPERIMENTATION

We discuss how theorem 2 compares with experimental determination of isophote curves on different objects. All the experimentation presented was done on real objects.

Figure 3a shows different isophotes, depicted in white, on the image of a glass bottle sprayed with diffuse white paint. The bottle is illuminated with a circular light source, with diameter about $1/4$ the length of the bottle, and placed about 4 bottle lengths away. The surface of the bottle consists of three 2-D regions of parabolic points. Starting from the bottom, the bottle consists of (i) a cylindrical region, and then (ii) a cone region with significant cone half-angle, and then towards the top, (iii) a narrower cone region. In figure 3a the isophotes clearly appear to follow along the principal direction of curvature zero in each of these regions. At the very bottom of the bottle there is a tapering inward which bends the isophote curves towards the horizontal.

Where the approximation to a distant light source becomes a problem for theorem 2, is for isophotes close to the specular highlight region on the lower cylindrical portion of the bottle. Ideally, for a light source placed far away from the bottle, the highlight region should appear as a linear stripe for a cylinder or cone region along points of equal surface orientation. This is true to good approximation for the faint highlight region near the top of the bottle where the non-zero curvature is high. However, for the cylindrical portion of the bottle where the non zero curvature is lower the highlight appears more elliptical due to the finite circular extent of the light source. Recall the discussion in section 3 regarding how the approximation to invariance of isophotes at parabolic points gets better for higher non-zero viewer centered curvature λ .

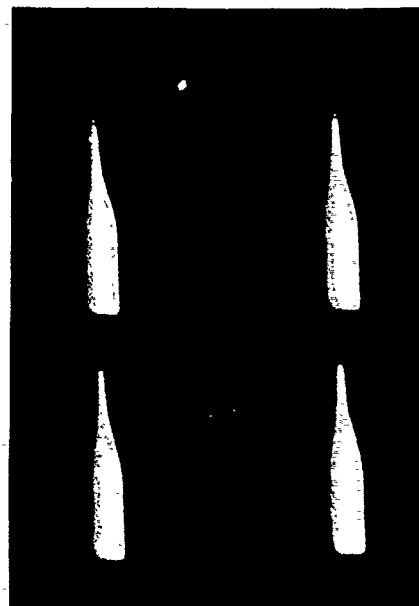
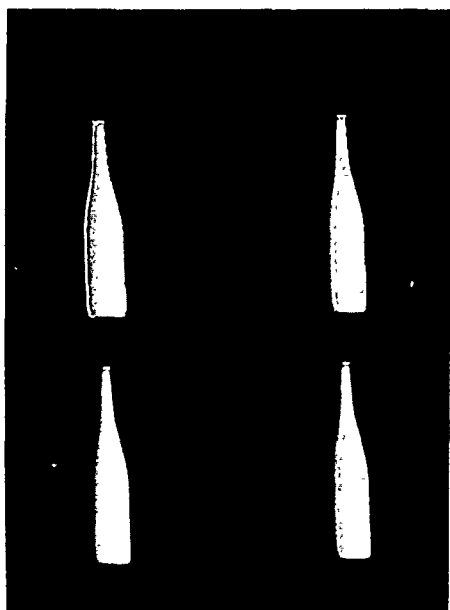


Figure 3:



Figure 4:

As depicted in figure 3b, isophotes become elliptically shaped very close to this highlight region. Figure 3b shows how well behaved isophotes, with respect to theorem 2, degenerate into tightly closed ellipses as they approach closer to the highlight on the cylindrical portion of the bottle. At least the major axis of the ellipses is aligned with the principal direction of zero curvature.

Figure 4 shows a perspective view of a number of cylindrical objects in various orientations illuminated by the same circular light source as in figures 3a and 3b. The non-zero principal curvature is large enough so that the distant light source approximation

is very good relative to its given finite distance of only about 2 feet from the objects. The linear specular highlights align themselves very well along the direction of zero curvature, and serve as a good cue for constraining their cylindrical orientation in 3-space. Note the intensity threshold segmentation image on the right.

Figure 5 shows the edge of a specular highlight on a torus for two different orientations in 3-space, defining its parabolic line (in this case a circular ring). The tangent direction to the parabolic line on a torus is the direction of zero curvature giving experimental evidence of theorem 2, in this case for a specular isophote.

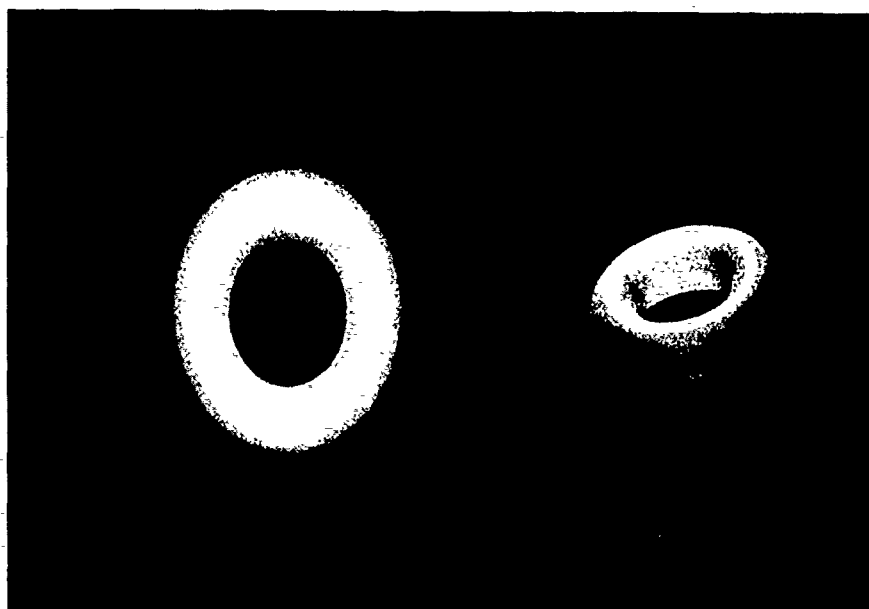


Figure 5:

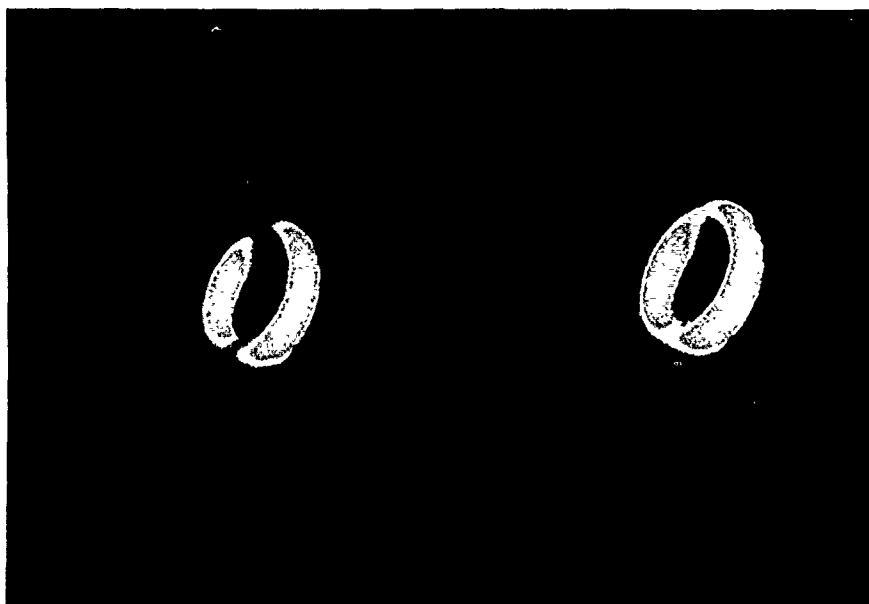


Figure 6:

Figure 6 shows diffuse isophotes on a torus as they approach the ring of parabolic points on a torus. The left image shows for a given gray value two disjoint isophote curves. As the gray value is increased, the disjoint isophote curves merge so that the top half of each disjoint curve form the parabolic ring on top of the torus (see right image of figure 6). Hence the tangent to the diffuse isophote curves at the parabolic points on the torus is parallel to the direction of zero curvature.

Figure 7 shows specularities produced from linear fluorescent lighting on a curved glass bottle. In spite of large albedo variations due to different intensities being transmitted through the partially transparent glass, the highlight curves delineate the direction of zero curvature within parabolic regions of the bottle which are cone shaped. Observe how some of the highlight curves get perturbed at non-parabolic points where the bottle bulges outward.



Figure 7:

5.2 CONSTRAINTS ON SURFACE ORIENTATION

With respect to the tangent plane at a point on a smooth surface, principal directions of curvature are orthogonal. If the projection of the principal directions of curvature onto the image plane are known, then a surface orientation constraint is produced from the skewed symmetry of these orthogonal directions. The direction of an isophote (or equivalently the direction of the image intensity gradient) serves as a cue for finding one of these projected principal directions of curvature at parabolic points. Such a cue can be useful in recovering the shape of cylinders with arbitrary smooth cross section.

[Kanade 1981] was the first to apply skewed symmetry constraints to understanding polygonal line drawings. Given a pair of directions depicted in figure 8a at angles α and β relative to the horizontal, if in 3-D space these directions are orthogonal, then [Kanade 1981] derives the following orientation constraint on the normal to the plane within which these directions lie relative to the image plane:

$$\cos(\alpha - \beta) + (p \cos \alpha + q \sin \alpha)(p \cos \beta + q \sin \beta) = 0. \quad (6)$$

This constraint equation produces a hyperbola of solutions in gradient coordinates depicted in figure 8b.

On a cylinder with arbitrary smooth cross section, all points are parabolic. From theorem 2 isophotes give a cue for the projected principal direction of zero curvature. The contour of the cross section gives a cue for the projected non-zero principal curvature direction. Therefore equation 6 is applicable to the tangents of intersecting cross section and isophote curves. If the reflectance map, $R(p, q)$, is known as a function of surface orientation then the image irradiance equation can further constrain surface orientation to a discrete number of points.

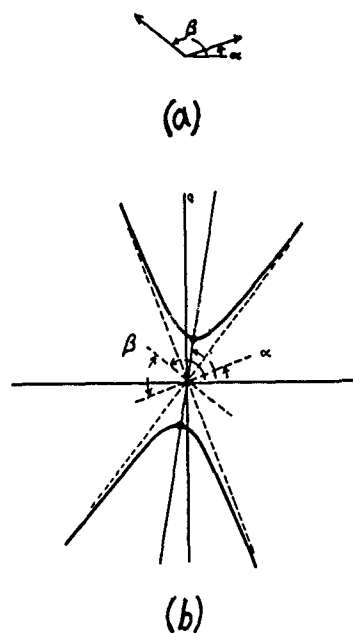


Figure 8:

Consider generalizing to right linear straight homogeneous generalized cylinders (e.g., cones with arbitrary smooth cross section), on which all points are also parabolic, but cross sectional contours are no longer necessarily lines of curvature. Isophotes serve as a cue for the direction of the meridians for such generalized cylinders. Such a cue can help generate a "net" which is an array of intersecting cross section and meridian curves. Using this "net" [Udupinar and Nevatia 1988] derive 3-D orientation constraints for the axis of certain right linear straight homogeneous generalized cylinders which have symmetric contour.

On right straight homogeneous generalized cylinders (SHGCs) (i.e., cylinders with arbitrary smooth cross section that can be arbitrarily scaled along a straight axis, see [Agin and Binford 1973], [Shafer 1985], [Ponce Chelberg and Mann 1989]), theorem 2 is applicable only at cross sections and meridians³ which are parabolic lines. [Ponce Chelberg and Mann 1989] show that parabolic points occur on occluding contour curves and cross section contour curves at inflection points of these curves. They then show a method for determining the projection of the straight axis of SHGCs onto the image plane. Using these methods of [Ponce Chelberg and Mann 1989], the paper by [Gross and Boulton 1989] shows how to generate all visible cross sections and meridians along an SHGC. This includes those cross sections and meridians that are parabolic lines which necessarily connect between points of inflection on contours.

What more can be learned about the 3-dimensional shape of an SHGC by observing isophote cues for principal directions of zero curvature at points along these parabolic lines (assuming these parabolic curves exist)? Relatively little work has been done on shape determination of SHGCs from principal directions of curvature, primarily because, in general, cross sectional and meridian curves are rarely principal lines of curvature (see [Brady et al. 1985]). However this newly discovered geometric property of isophotes at parabolic points supplies more motivation to study shape determination of SHGCs from knowledge of certain principal directions of curvature.

5.3 DETERMINING CURVATURE COMPONENTS AT A PARABOLIC POINT

We have already encountered 3 equations constraining surface geometry from radiometric measurement, the image irradiance equation using an image irradiance value:

$$I(x, y) = R(p, q)$$

and equations 1 using the components of the image gradient:

$$I_x = R_p \partial p / \partial x + R_q \partial q / \partial x \quad I_y = R_p \partial p / \partial y + R_q \partial q / \partial y \quad (7)$$

Here we have assumed symmetry of the viewer centered curvature matrix so that $\partial q / \partial x = \partial p / \partial y$. Determining the image direction, (a, b) , of zero curvature at a parabolic point introduces two linear equations:

$$\begin{pmatrix} \partial p / \partial x & \partial q / \partial x \\ \partial q / \partial x & \partial q / \partial y \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (8)$$

At first it may appear that we have a chance at a discrete local solution, at parabolic points, to the shape from shading problem in 5 variables;

$$p, q, \partial p / \partial x, \partial q / \partial x, \partial q / \partial y.$$

However theorem 2 tells us that the ratio of a and b are constrained by the image gradient as follows:

$$\frac{b}{a} = -\frac{I_x}{I_y} \quad (9)$$

In fact it is the image gradient in the first place that gives us the directional cue for zero curvature at parabolic points.

Even though the 4 linear equations 7 and 8 have some dependency upon each other, there is enough constraint information to determine curvatures $\partial p / \partial x, \partial q / \partial x, \partial q / \partial y$ under certain conditions. Together with the constraint equation 9 they determine the equation

$$I_z = \partial p / \partial x (R_p - \frac{a}{b} R_q) \quad (10)$$

Clearly if surface orientation is known, as well as the reflectance map $R(p, q)$, the quantity $\partial p / \partial x$ can be uniquely determined. From equations 8 the quantities $\partial q / \partial x$ and $\partial q / \partial y$ can be subsequently determined.

If surface orientation is not known *a priori*, curvature can be determined up to a discrete set of solution points if the projection of the non-zero principal direction of curvature is known onto the image plane. We discussed in section 5.2 how surface orientation in this case is constrained by the skew symmetry equation 6, and how together with the image irradiance equation surface orientation is determined up to a discrete set of points. Substituting these orientation measurements into equation 10 yields a discrete set of curvature measurements.

6 CONCLUSION

In this paper we have carefully considered the invariance of the direction of isophotes (equivalently the image intensity gradient) at parabolic points on smooth surfaces with respect to all smooth reflectance functions. While other researchers have considered this photometric invariance property as a unique characteristic of parabolic points that could lead to their identification on smooth objects, we have considered this photometric invariance property, in terms of how it relates to the geometry of surfaces at parabolic points. Using rigorous methods we have shown that the invariant direction of isophotes at parabolic points is the projection of the principal direction of zero curvature onto the image plane. We have gone beyond just considering the limited case of pure orthographic projection, and have shown under what conditions this photometric invariance property breaks down and is best approximated in perspective images, and for a nondistant light source.

In addition to giving experimental evidence supporting our claim about how isophotes are parallel to the direction of zero curvature at parabolic points, we suggest possible applications of this geometric cue to determination of orientation and curvature at parabolic points. The best advantage taken from this geometric cue is when constraint information can be obtained, knowing absolutely nothing about the reflectance map. The true power of a physical invariant is the information it gives under the most generic conditions possible, in this case any type of smooth reflectance map. Perhaps there are other physical invariants awaiting to be discovered in vision.

References

- [Agin and Binford 1973] Agin, G.J., and Binford, T.O., *Computer description of curved objects*, Proceedings of the 3rd IJCAI, Stanford California, pp. 629-640, 1973.
- [Babu Lee and Rosenfeld 1985] Babu, M.D.R., Lee, C.-H., and Rosenfeld, A., *Determining plane orientation from specular reflectance*, Pattern Recognition, 18(1): pp.53-62, 1985.

³In this general case, a meridian is a tracing produced by a point on a cross section as it is being swept over the straight line axis and scaled

[Besl and Jain 1985] Besl, P., and Jain, R., *Intrinsic and Extrinsic Surface Characteristics*, Proceedings of IEEE conference on Computer Vision and Pattern Recognition (CVPR) 1985, pp.226-233.

[Blake Zisserman and Knowles 1985] Blake, A., Zisserman, A., and Knowles, G., *Surface Descriptions from Stereo and Shading*, Image and Vision Computing, Vol. 3, No. 4, pp. 183-191,

[Brady et al. 1985] Brady, M., Ponce, J., Yuille, A., and Asada, H., *Describing Surfaces*, in Proceedings 2nd International Symposium Robotics Research, Harasuja and Inoue, Eds. Cambridge, MA, MIT Press, 1985.

[Brelstaff 1989] Brelstaff, G.J., *Inferring Surface Shape From Specular Reflections*, Phd Thesis, Department of computer science, University of Edinburgh, 1989.

[Buchanan 1987] Buchanan, C.S., *Determining Surface Orientation From Specular Highlights*, RCBV-TR-87-19, University of Toronto, Masters Thesis, August 1987.

[Cernuschi-Frias and Cooper 1984] Cernuschi, B., and Cooper, D.B., *3-D Space Location and Orientation Parameter Estimation of Lambertian Spheres and Cylinders From a Single 2-D Image by Fitting Lines and Ellipses to Thresholded Data*, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 6, No. 4, pp. 430-441, 1984.

[do Carmo 1976] do Carmo, M.P., *Differential Geometry of Curves and Surfaces*, Prentice Hall, 1976.

[Gross and Boulton 1989] Gross, A., and Boulton, T., *Recovery of Straight Homogeneous Generalized Cylinders Using Contour and Intensity Information*, SPIE, Visual Communications and Image Processing IV, pp. 1661-1669, Vol. 1199, Nov. 1989.

[Healey and Binford 1987] Healey, G., and Binford, T.O., *Local Shape From Specularity*, Proceedings of the First International Conference on Computer Vision (ICCV), pp.151-160, 1987.

[Horn 70] Horn, B.K.P., *Shape from Shading: A Method for Obtaining the Shape of a Smooth Opaque Object from one view*, PhD Thesis, Department of Electrical Engineering, MIT.

[Horn and Sjöberg 1979] Horn, B.K.P., and Sjöberg, R.W., *Calculating the Reflectance Map*, Applied Optics, Vol. 18, No. 11, pp. 1770-1779, June 1979.

[Hildebrand 1965] Hildebrand, F.B., *Methods of Applied Mathematics*, Prentice Hall, Englewood Cliffs, N.J., 1965.

[Kanade 1981] Kanade, T., *Recovery of the 3-D Shape of an Object From a Single View*, Artificial Intelligence, Vol. 17, pp. 409-460, August 1981.

[Koenderink and van Doorn 1980] Koenderink, J., and van Doorn, A.J., *Photometric Invariants Related to Solid Shape*, Optica Acta, Vol. 27, No. 7, pp. 981-996, 1980.

[Lee and Rosenfeld 1985] Lee, C.H., Rosenfeld, A., *Improved Methods of Estimating Shape from Shading Using the Light Source Coordinate System*, Artificial Intelligence, Vol. 26, No. 2, pp. 125-143, 1985.

[Pentland 1984] Pentland, A., *Local Shading Analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 6, No. 2, pp. 170-187, 1984.

[Ponce Chelberg and Mann 1989] Ponce, J., Chelberg, D., and Mann, W.B., *Invariant Properties of Straight Homogeneous Generalized Cylinders and Their Contours*, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 12, No. 9, pp. 951-966, 1989.

[Shafer 1985] Shafer, S.A., *Shadows and Silhouettes in Computer Vision*, Kluwer Academic Publishers, Boston, 1985.

[Thrift and Lee 1983] Thrift, P., and Lee, C., *Using Highlights to Constrain Object Size and Location*, IEEE Trans. Syst., Man, Cybern., SMC-13, pp.426-431, 1983.

[Torrance and Sparrow 1967] Torrance, K. and Sparrow, E., *Theory for Off-Specular Reflection from Roughened Surfaces*, Journal of the Optical society of America, 57, pp.1105-1114, 1967.

[Ulupinar and Nevatia 1988] Ulupinar, F., and Nevatia, R., *Using Symmetries for Analysis of Shape from Contour*, Proceedings of the Second International Conference on Computer Vision (ICCV), pp.414-426, 1988.

[Woodham 1978] Woodham, R.J., *Reflectance map techniques for analyzing surface defects in metal castings*, MIT AI Lab Tech Report AI-TR-457, June 1978.

APPENDIX

We give a more formal quantitative proof of theorem 1 which involves precisely relating the surface centered curvature matrix to the viewer centered curvature matrix. A number of concepts from differential geometry are used, and specific page numbers in [do Carmo 1976] are referenced where they are defined. [Besl and Jain 1985] is also a good reference which surveys most of these concepts. Again, we use the same style notation as in [do Carmo 1976].

For a surface, $S(x, y)$, parameterized by height with respect to image coordinates (x, y) , we can easily relate image coordinates to a set of local coordinates for each point on the surface. Since image pixels are usually square we can consider (x, y) to be an orthogonal image coordinate system. The constant lines of x and y projected onto the surface, $S(x, y)$, are called the *parallels* for the surface with respect to the coordinatization of the surface by (x, y) (see figure 9). The parallels of S are therefore surface

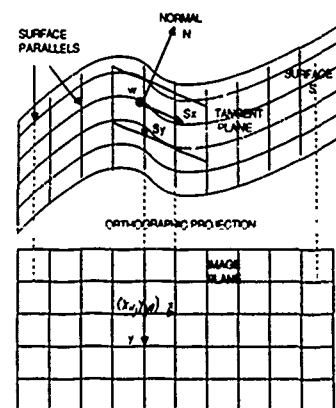


Figure 9:

curves parameterized by lines of constant x or y . At a point, w , on the surface, consider the unique two intersecting parallels, one parallel parameterized by an image line of constant x and the other parallel parameterized by an image line of constant y . Consider the tangent vectors S_x and S_y at the point w . Since the surface is a height parameterization,

$$S(x, y) = (x, y, f(x, y)),$$

where $f(x, y)$ is called the *height function*. Therefore

$$S_x = (1, 0, p), \quad S_y = (0, 1, q) \quad (11)$$

where we are using the gradient coordinates p and q defined by

$$p = \partial f / \partial x, \quad q = \partial f / \partial y.$$

The tangent vectors S_x and S_y at w form a local coordinate system. At image coordinates (x_w, y_w) lying orthographically beneath w , the local displacement in image coordinates (a, b) corresponds to the local displacement

$$aS_x + bS_y = (a, b, ap + bq)$$

in local surface coordinates at point w . Therefore the relationship between local image coordinates and local surface coordinates with respect to corresponding image coordinate tangents with surface parallel tangents is quite simple; local image coordinates (a, b) correspond to local surface coordinates (a, b) , just as long as it is kept in mind that these representations are with respect to image coordinate tangents and surface parallel tangents, respectively.

The Gauss map with respect to the surface, S , maps each point, w , onto the point on the unit sphere with local surface normal parallel to the normal at w on S . Using equations 11 the unit normal at point w is

$$N = \frac{S_y \times S_x}{\|S_y \times S_x\|} = \left(\frac{p}{\sqrt{1+p^2+q^2}}, \frac{q}{\sqrt{1+p^2+q^2}}, \frac{-1}{\sqrt{1+p^2+q^2}} \right). \quad (12)$$

The *differential* of the Gauss map relates the rate of change of the surface normal at point w on the surface S with respect to local vector displacements in the tangent plane at point w ([do Carmo 1976] pp.135-137). The differential of the Gauss map can be represented by a 2x2 matrix which multiplies vectors in the tangent plane to produce the vector displacement of the local surface normal parallel to the tangent plane. In this paper we term the differential of the Gauss map the *surface centered curvature matrix*. The Gaussian curvature is the determinant of this matrix and the principal directions are the respective eigenvectors.

A representation of the 2x2 surface centered curvature matrix in local surface coordinates using surface parallel tangents, is given by the *equations of Weingarten* ([do Carmo 1976] pp.153-155). For a general parameterization of a surface, the surface centered curvature matrix in surface parallel tangent coordinates is

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

where

$$a_{11} = \frac{fF - eG}{EG - F^2}, \quad a_{12} = \frac{gF - fG}{EG - F^2},$$

$$a_{21} = \frac{eF - fE}{EG - F^2}, \quad a_{22} = \frac{fF - gE}{EG - F^2},$$

where E, F, G are coefficients for the first fundamental form and e, f, g are coefficients for the second fundamental form. The equations of Weingarten can be easily derived for the height parameterization by computing the first and second fundamental forms for this parameterization.

The first fundamental form ([do Carmo 1976] p.92) is computed from the following inner products:

$$E = \langle S_x, S_x \rangle = 1 + p^2, \quad F = \langle S_x, S_y \rangle = pq,$$

$$G = \langle S_y, S_y \rangle = 1 + q^2.$$

The second fundamental form ([do Carmo 1976] p.141) is computed from a set of inner products involving the rate of change of the surface normal, N . Using equation 12:

$$e = \langle N_x, S_x \rangle = \frac{\partial p / \partial x}{\sqrt{1+p^2+q^2}}, \quad f = \langle N_x, S_y \rangle = \frac{\partial q / \partial x}{\sqrt{1+p^2+q^2}},$$

$$g = \langle N_y, S_y \rangle = \frac{\partial q / \partial y}{\sqrt{1+p^2+q^2}}.$$

The derivation of the surface centered curvature matrix in [do Carmo 1976 p.155], using local surface coordinates, involves a useful decomposition of the surface centered curvature matrix into the product of the following two matrices:

$$\begin{pmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{pmatrix} = - \begin{pmatrix} E & F \\ F & G \end{pmatrix}^{-1} \begin{pmatrix} e & f \\ f & g \end{pmatrix}.$$

That is, the surface centered curvature matrix is the product of the inverse of the first fundamental form matrix, with the second fundamental form matrix.

For the height parameterization:

$$\begin{pmatrix} E & F \\ F & G \end{pmatrix}^{-1} = \frac{1}{1+p^2+q^2} \begin{pmatrix} 1+q^2 & -pq \\ -pq & 1+p^2 \end{pmatrix}$$

$$\begin{pmatrix} e & f \\ f & g \end{pmatrix} = \frac{1}{\sqrt{1+p^2+q^2}} \begin{pmatrix} \partial p / \partial x & \partial q / \partial x \\ \partial q / \partial x & \partial q / \partial y \end{pmatrix}.$$

Note that the second fundamental form matrix is simply a scalar multiple of the viewer centered curvature matrix in equation 1. Hence,

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = \underbrace{\frac{-1}{(1+p^2+q^2)^{3/2}} \begin{pmatrix} 1+q^2 & -pq \\ -pq & 1+p^2 \end{pmatrix}}_T \begin{pmatrix} \partial p / \partial x & \partial q / \partial x \\ \partial q / \partial x & \partial q / \partial y \end{pmatrix}. \quad (13)$$

The matrix T transforms the viewer centered curvature matrix into the surface centered curvature matrix, under matrix multiplication. The matrix T represents the transformation of gradient coordinates into local tangent plane coordinates. T is negative definite at non-occluding points since

$$DET(T) = \frac{-(1 + p^2 + q^2 + 2p^2q^2)}{(1 + p^2 + q^2)^{3/2}}.$$

Therefore a vector v gets multiplied to zero by the surface centered curvature matrix if and only if the vector v gets multiplied to zero by the viewer centered curvature matrix. Theorem 1 follows from the discussion of the geometry in figure 2.

Image Blurring Effects Due to Depth Discontinuities

Thang C. Nguyen *

Beckman Institute Rm 1614
405 N. Mathews St.
Urbana, IL 61801

Thomas S. Huang

Coordinated Sciences Laboratory, and
Beckman Institute
405 N. Mathews St.
Urbana, IL 61801

Abstract

A new model (called multi-component blurring, or MCB) to account for image blurring effects due to *depth discontinuities* is presented. We shows that blurring processes operating in the vicinity of large depth discontinuities can give rise to new image details that can be quite distinguishable and that cannot be explained by previously available blurring models (which can only predict suppression, and not creation, of image details.) Extensive and carefully constructed experiments to obtain images of real scenes taken by a CCD camera with typical parameters, point towards the qualitative validity of our new blurring model. Due care has been taken to ensure that the image phenomena observed are mainly due to blurring and not due to mutual illumination, specularity, objects' "finer" details, coherent diffraction, or incidental image noises.

Section 2 discusses the radiometry of image formation in the presence of sharp discontinuities. The object-and-camera configuration used in this analysis is also adapted to the real-scene experiments presented in section 3.

Section 3 presents the results from extensive experiments with images of realistic scenes taken with a Cohu-4815 CCD camera with 8-bit accuracy. Various settings of camera parameters (focal length f , aperture A and back-focal distance v) are employed. Also, controlled experiments for background checking were performed to ensure valid data. Though no quantitative analysis was conducted, due to the unavailability of radiometric calibration equipment at our Robot Vision Lab, we were nevertheless able to apply the MCB (multi-component blurring) model to *predict qualitatively* all of the various image blurring instances that were observed.

Section 4 discusses some implications of the MCB effects, especially towards applications such as depth-from-sharpness and other radiometric stereo methods as well as deblurring for images of close-range, high-relief (large depth range) scenes. Finally, further directions for research on multi-component blurring effects are outlined.

1 Introduction

The objectives of this paper are: to present a simplified image blurring model that is sufficiently general to account for blurring effects due to depth discontinuities, and to show that blurring in images of realistic 3-D scenes can be quite interesting and detectable. Experimental results are also presented and discussed.

The original motivation was just to find a simple model to describe image blurring so that depth-from-sharpness methods can be improved, especially in regions near depth discontinuities, since none of the depth-from-sharpness formulation so far has discuss such important cases in details. See [5-10]. Upon analyzing the blurring phenomena, it was realized that such a model must be composite (ie. consisting of a (possibly unknown) number of sub-processes.) And it is precisely the composite nature of blurring due to depth-discontinuities that the net blurring effects can be quite interesting, with new local extrema generated, entirely in *discord* with commonly employed blurring models (even those allowing a shift-varying kernel.) Even if each blurring sub-process is very simple, (shift-invariant Gaussian blurring, for example), the *composite* kernel can be quite complex, exhibiting discontinuities, and is strongly shift-varying.

The organization of this paper is as follows:

* Support from the National Science Foundation, through the Engineering Creativity Award EID-8811553, is gratefully acknowledged.

2 Modeling of Multi-component Blurring

In this section, we consider a particular camera configuration to study blurring in the presence of depth discontinuities. The setup is chosen to be amenable to 1-dimensional analysis for clarity of discussion only, since extension to two dimensions is straightforward.

The width of a blurring kernel can be defined as: (similar to Subbarao [6])

$$w_{K_i} \triangleq \frac{\int_{-\infty}^{\infty} (x - \bar{x})^2 K_i(x) dx}{\int_{-\infty}^{\infty} K_i(x) dx}; \quad \bar{x} = \frac{\int_{-\infty}^{\infty} x K_i(x) dx}{\int_{-\infty}^{\infty} K_i(x) dx} \quad (1)$$

related to:

$$w_{K_i} \simeq kD \frac{f}{u_i - f}; \quad \text{when } u_0 \gg u_i \quad (2)$$

so setting the focus to very far away from camera is a desirable technique for depth-from-sharpness to prevent ambiguities. We will assumed such a setting in discussing MCB below.

2.1. Image formation across depth discontinuities

Refer to figure 1 for the subsequent discussions in this section.

Figure 1 represent a simplified model of a camera imaging a sharp edge that "touches" on the optical axis and lying in front of a uniform background. The Lambertian assumption, though convenient, is not required here, only that no specular reflection is registered onto the image, and that no significant mutual illumination effects (interreflections) are present near the points of interest. Care must be exercised to prevent the aforementioned effects since they sometimes also create spurious image features that can be confused with multi-component blurring effects [1, 2].

Quantities related to the background are subscripted with a B (or b). Those related to the edge by E or e . The separate image components due to background and edge are analyzed independently. We will concentrate on the cases where one of the blurring process is *dominant* (ie. having a much larger spread than others) in the image neighborhood. (In figure 1 the blurring due to E is dominant.)

2.1.1. Notations and configuration

- The back ground B is assumed to be in better focus (closer to the best-focus plane) than the edge E . As explained earlier, in depth-from-sharpness systems, it is convenient to have the camera best focused at very far, so that closer objects are more blurred.
- The unblurred images of the background patch B and edge E are $I_{e0}\xi(x)$, and $I_{b0}\xi(-x)$, disjoint half lines, or half plane in 2-D images. This is taken so that the emergence of any new details due to blurring at the "interface" ($x = 0$) are readily observable and not shifted due to change in magnification factor due to camera parameters so that direct interpretations on real image below is straightforward.
- Due to the fact that w_b varies along the (image) x -axis, (due to occlusion), the blurring kernel $K_b(x, x')$ is shift-varying, despite the fact that the depth value across the background patch is constant. However, since w_b is taken to be negligible compared to w_e , the exact

behavior of $K_b(x, x')$ is not important, hence in the next step of modeling, we will assume a shift-invariant Gaussian form for both $K_e(x)$, $K_b(x)$.

2.1.2. Image formation and blurring effects

For the one-dimensional model in figure 1, we can see that, at each image coordinate value x , the resulting intensity also contain the sum of all blurring (or diffusion) contribution from neighboring image regions (ie. pixels, etc.), each of which may have a different blurring kernel. Concisely, then:

$$I(x) = \sum_{j \in J} I_j(x);$$

j indexes the set of different blurring kernels that have significant contribution at x (3)

for our 2-component case, in particular, we have:

$$I(x) = I_e(x) + I_b(x) : 2 \text{ components}$$

$$I_e(x) = \int_0^{\infty} K_{we}(x, x') I_{e0} T_e(x') dx' \quad (4)$$

$$I_b(x) = \int_{-\infty}^0 K_{wb}(x, x') I_{b0} T_b(x') dx'$$

where:

- $I_e(x)$ is the blurred image component for edge E .
- The operator (blurring kernel) that give $I_e(x)$, from an ideal radiometric image of edge E ($I_{e0}\xi(x)$), is modeled approximately by a shift-invariant $K_{we}(x-x')$, with width w_e . (Later, a shift-invariant Gaussian $G_{\sigma_e}(x)$ is used in next stage of modeling simplification.)
- $T_e(x)$ is the lens interception function that describes how much image intensity results from a particular direction.

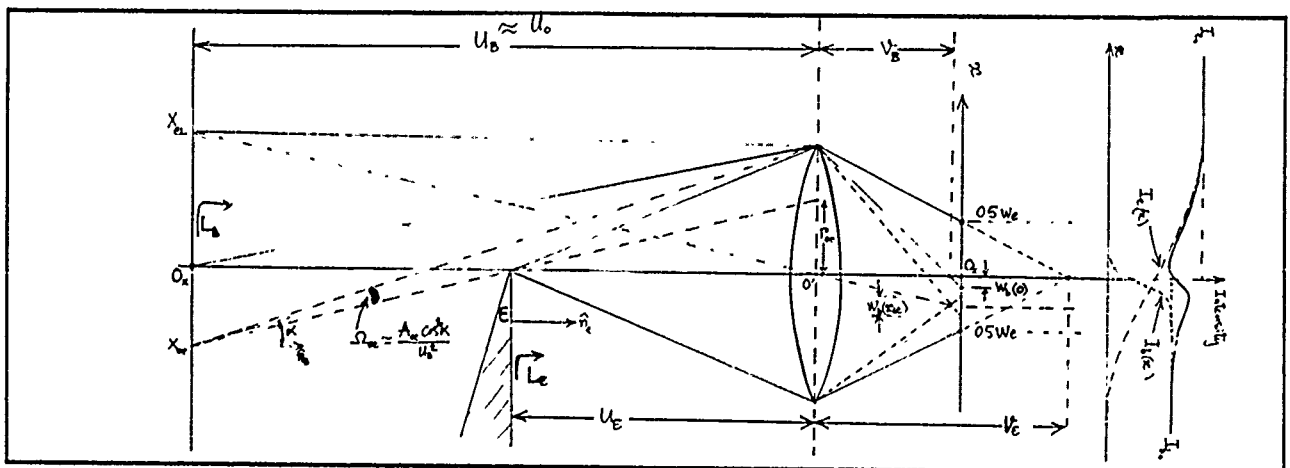


Figure 1. Imaging a sharp discontinuity

From the above rather general model, we will make the following further simplifications to facilitate discussions on the MCB effects. The Gaussian shape with shift-invariant widths are subsequently assumed for all blurring kernels. This will hold well only for the *dominant* blurring kernel (here is $K_e(x)$, corresponding to the surface patch closest to the camera, with largest blur width), but this is precisely what we look for here, since the closest surface is usually of immediate interest in navigation, picking, etc.

These assumptions enable an approximate analytical representation of multi-component blurring with Gaussian kernels in the next section. With the analogy between Gaussian blurring and heat diffusion [3, 4], multi-component blurring is analogous to multi-component particle diffusion, whereas each type of particle has a different diffusion constant, and none of them react chemically with each other. Note that MCB is more complex than anisotropic diffusion [4, 14].

2.2. Creation of image details by Multi-component Gaussian Blurring Effects

Now, we want to show that multi-component blurring can give rise to *new image features* (or details), by which we mean specifically new local extrema, ie. local peaks and valleys. These are relatively simple to represent mathematically for detection schemes, and also readily observed visually. Please also refer to figure 2 and figure 3 (a through d), for the following discussion.

The 1-dimensional unblurred image is again taken to be two disjoint step functions with heights I_{a0} , I_{b0} . The Gaussian-blurred components are $I_a(x)$, $I_b(x)$ respectively.

$$I_{\text{unblurred}}(x) = I_{a0}\xi(x) + I_{b0}\xi(-x);$$

$$I(x) = I_a(x) + I_b(x);$$

$$I_a(x) = G_a(x) * I_{a0}\xi(x); \quad I_b(x) = G_b(x) * I_{b0}\xi(-x)$$

$$\xi(u) = \begin{cases} 1; & u \geq 0 \\ 0; & u < 0 \end{cases} \quad \text{unit step function} \quad (5)$$

Where $*$ denotes convolution. We are interested in finding the conditions that leads to resulting images like those in figures 3b, 3c and 3d (new local extrema created). Note that with ideal step functions as shown, any continuous, uni-modal single blurring kernel will not introduce new details, like new local extrema. (This is the main assumption in the Gaussian scale-space concept, however, MCB does not obey such restrictions.) Let's look at the total image's derivative:

$$\begin{aligned} \frac{\partial}{\partial x} (G_a(x) * I_{a0}\xi(x) + G_b(x) * I_{b0}\xi(-x)) \\ = I_{a0}G_a(x) - I_{b0}G_b(x) = 0; \\ \text{or } \frac{1}{\sqrt{2\pi}} \left(\frac{I_{a0}}{\sigma_a} \exp\left(\frac{-x_z^2}{2\sigma_a^2}\right) - \frac{I_{b0}}{\sigma_b} \exp\left(\frac{-x_z^2}{2\sigma_b^2}\right) \right) = 0; \\ \text{or } \frac{I_{a0}\sigma_b}{I_{b0}\sigma_a} = \exp\left(\frac{x_z^2}{2}\right) \left(\frac{1}{\sigma_a^2} - \frac{1}{\sigma_b^2} \right); \end{aligned} \quad (6)$$

The resulting conditions and solution for x_z , coordinate of local extrema formed by MCB blurring, are:

$$\begin{aligned} \text{if } (\sigma_a > \sigma_b) \text{ and } \left(\frac{I_{a0}}{\sigma_a} < \frac{I_{b0}}{\sigma_b} \right); \\ \text{or if } (\sigma_a < \sigma_b) \text{ and } \left(\frac{I_{a0}}{\sigma_a} > \frac{I_{b0}}{\sigma_b} \right); \end{aligned}$$

$$\text{which gives } \text{abs}(x_z) = \sigma_a\sigma_b \sqrt{\frac{2 \ln \left(\frac{I_{a0}\sigma_b}{I_{b0}\sigma_a} \right)}{(\sigma_b + \sigma_a)(\sigma_b - \sigma_a)}} \quad (7)$$

Physical limitations such as blooming and smear of the imaging sensor (pixels) [13] and the Optical Transfer Function (OTF) [11, 12] of the lens system means that there is an upper limit to I_{a0}/I_{b0} above. That is, charge spilling between adjacent pixels, prevents too large a charge difference between neighbors. Similar limitation due to roll-off of the (OTF) also play a role. The net effect is that the local extrema by MCB are detectable for some range of I_{a0}/I_{b0} , with some upper limits dictated by CCD sensor characteristics, and lower limit at least as high as given by equation (10). This suggests that MCB effects is more detectable at low local contrast, a rather surprising prediction that was observed in real images. See figures 3 to 4.

If one try, for example, to describe the image profiles 3a, 3b, 3c and 3d by the convolution of a single kernel with the total unblurred profile (a step edge): the resulting kernel $K_{\text{composite}}(x, x')$ must be given by:

$$K_{\text{composite}}(x, x') = \begin{cases} G_a(x - x'), & x' \geq 0 \\ G_b(x - x'), & x' < 0 \end{cases} \quad (8)$$

which looks innocuously simple, until we see some sample plots of it in figures 3a, 3b. As seen, with $\sigma_a=3$, $\sigma_b=5$, $K_{\text{composite}}(x, x')$ is neither Gaussian (it's a patching of 2 truncated Gaussian segments), nor shift-invariant, and not even continuous at $x'=0$. Though this case is rather special, we can readily appreciate that blurring in high-relief 3-D scenes can be a complicated process to estimate, because even for shift-invariant Gaussian blurring we cannot get exact inverse solution (ie. for de-blurring or estimation of the blur width.)[3]

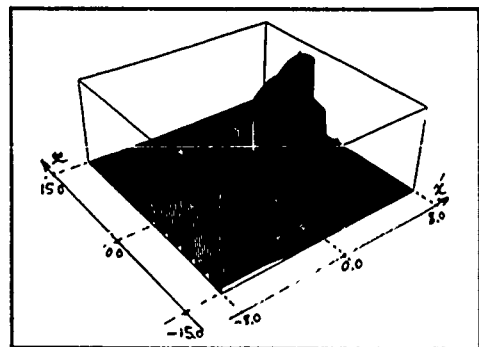


Figure 2. Composite kernel

3 Investigation of MCB Effects

in Real Images

From the above model for MCB, we set out to experiment with real images to test the hypothesis that MCB effects do exist and can be detected in images of realistic scenes.

3.1. Experimental setup

The set up is quite similar to the imaging model in figure 1. Distances from camera are: 1.33 meters to edge of board (E), and 5.69 meters to the 3 cardboards that served as background (B) on the wall. The camera optical axis makes angles of about 10.3 degrees and 12.5 degrees respectively to the surface normals of the edge E and background B, in opposite orientations to prevent "details" by specular reflections. We have insisted that other phenomena different than MCB are excluded from registering onto the images, by providing indirect, diffuse and non-polarized lighting and well-separated surfaces [14]. Coherent diffraction blurrings in our case are also limited to less than a pixel's width, so negligible to phenomena discussed here [13, 14].

3.2. Data (real-images) collection and interpretation

3.2.1. Image data

Almost a thousand images have been taken, with tens of different scenes giving consistent results. Here we included only two typical sets of images and their video scan lines for further discussions. See figure 3 and 4. There are a few interesting points to note:

- Multi-component blurring effects are more detectable at low local contrast, other factors unchanged.
- A small difference of blurring widths (due to different depths) could be detected and estimated.

Figure 3 illustrates the effects of local contrast across the depth-discontinuities, seen by varying the lighting on the front surface (board edge) and background independently. To test the MCB model validity, the real image profiles have been superimposed on the theoretical MCB profiles, which are based on separate estimations of blur widths for fore- and background by method of Subbarao [6]. Blur width of front edge estimated at 2.081 pixel widths, or about $23.9 \mu\text{m}$, while that of background is about 1.666 pixels, in figure 4.

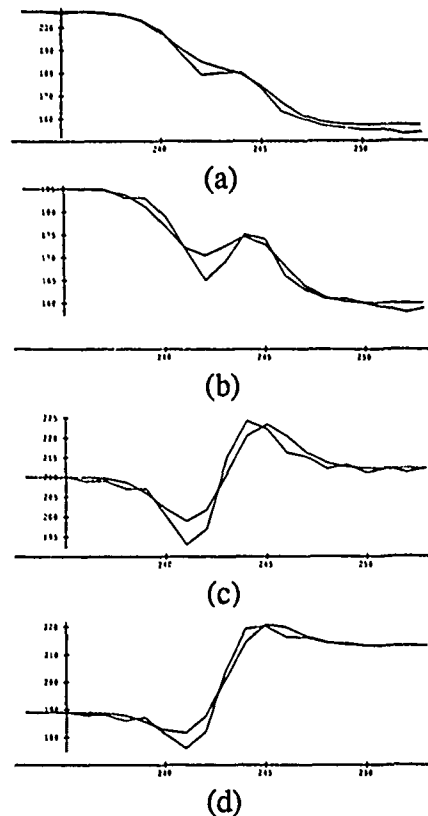
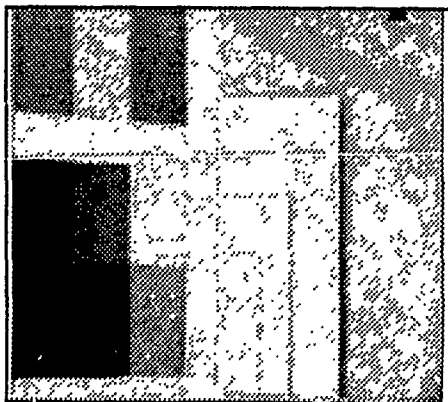


Figure 3. Contrast effects

Only a middle segment of the image is shown in figure 4. Figure 4a has blur widths of 3.919 pixels and 3.454 pixels respectively for the fore- and background. Figure 4b has 3.233 pixels and 2.622 pixels for the respective blur widths. Camera aperture is reduced from figure 4a to figure 4b. Note the reduce in overall image brightness from 4a to 4b, and the vertical blurring streaks in both 3 and 4.

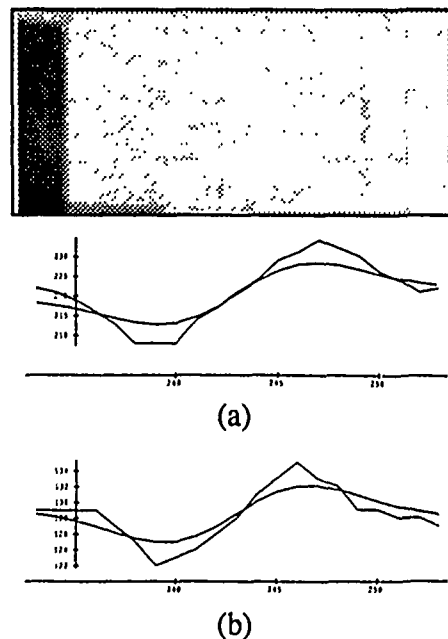


Figure 4. Effects of camera parameters

3.2.2. Interpretations

Pair-wise comparisons of slices 3a with 3c, and 3c with 3d: reduced local contrast enhances the "MCB spike"! This could imply that human depth-perception may be enhanced *for free* by MCB in low-contrast, large depth-range scenes.

The rough fitting of theoretical and actual image profiles above points towards the coarse accuracy of our current model. More refinements taking into account effects of occlusions are currently investigated by us [14]. No other blurring models so far has been able to describe the profiles in figures 3 and 4 above, however.

A side remark: the magnitude of the MCB peaks and troughs are significant, despite their "dull" appearance in the plots, which have been stretched out for detailed comparisons between theoretical MCB and actual image profiles. Excursions (from ideal unblurred edge profile) of ten or more gray level units are all present in the plots, and they are not rare in real images. However, their similarity to the Mach-band effects in human perception could have led people to dismiss MCB effects as visual illusions (which MCB is *not*). But not all cases of MCB are similar to Mach-band effects (see figure 3b.)

4 Discussions and Conclusions

An immediate implication from MCB is that simple depth-from-sharpness algorithms that rely on a single blurring kernel with *invariant shape* can give large errors in depth estimates for regions that are images of depth-discontinuities. It appears that current depth-from-sharpness formulations either have not taken account of effects due to MCB blurring [5, 6, 7, 8, 9], or *unknowingly* treat them as noise [8, 9, 10] (by spatial pre-averaging of intensity and post-averaging of depth estimates.) Such practices will give inaccurate depth estimates (averaging of estimates mainly serves redistribution of errors) and unknowingly *discard valuable depth information* carried in these image features.

The presence of MCB effects in images of 3-D scenes with large depth discontinuities suggests that de-blurring algorithm to be applied to such images may be improved by incorporation of some appropriate model for MCB. Whether or not such a refinement is necessary and/or desirable generally depends both on the scene characteristics (local contrast and depth variation), and results presented here can help in such decision-making.

We venture to speculate that MCB effects could have been active in helping our depth-perception in environments of low-contrast and large depth variations (dangerous environments indeed), thus such useful built-in capability of

MCB recognition through evolution could be possible [12, 13]. However, we would have to wait for any in-depth psychophysical investigation into the human perception of multicomponent blurring for definite results [14].

References

- [1] Forsyth, D., and Zisserman, A., "Mutual Illuminations", *Proc. Computer Vision and Pattern Recognition* (1989), California, USA, pp. 466-473.
- [2] Healey, G. and Bindford, T., "Local Shape from Specularity", *Proc. of the 1st Intl. Conf. on Computer Vision (ICCV'87)*, London, UK, (1987), pp 151-160.
- [3] Hummel, R., Kimia, B. and Zucker, S., "Gaussian Blur and the Heat Equation: Forward and Inverse Solution", *Proc. Computer Vision and Pattern Recognition* (1985), pp. 668-671.
- [4] Perona, P. and Malik, J., "Scale-space and Edge Detection using Anisotropic Diffusion", *IEEE Workshop on Computer Vision* (1987), pp. 16-22.
- [5] Pentland, A., "A New Sense for Depth 12, 13] of Field", *IEEE Trans. on Pattern Recognition and Machine Intelligence*, Vol. PAMI-9, No. 4 (1987), pp. 523-531.
- [6] Subbarao, M., Depth Recovery from Blurred Edges, *Proc. Computer Vision and Pattern Recognition* (1988), pp. 498-503.
- [7] Grossman, P., "Depth from Focus", *Pattern Recognition Letters*, 5 (1987), pp. 63-69.
- [8] Darrell, T. and Worn K., "Pyramid-Based Depth from Focus", *Proc. Computer Vision and Pattern Recognition* (1988), pp. 504-509.
- [9] Pentland, A., Darrell, T., Turk, M., and Huang, W., "A Simple, Real-time Range Camera", *Proc. Computer Vision and Pattern Recognition* (1989), pp. 256-261.
- [10] Garibotto, G. and Storace, P. "3-D Range Estimate from the Focus Sharpness of Edges", *Proc. of the 4th Intl. Conf. on Image Analysis and Processing* (1987), Palermo, Italy, Vol. 2, pp. 321-328.
- [11] Ghatak, A. and Thyagarajan, K., *Contemporary Optics*, Plenum Press, New York, 1978.
- [12] Levine, M., *Vision in Man and Machine*, McGraw-Hill, 1985.
- [13] Texas Instruments Inc., "Advance Information Document for TI Imaging Sensor TC241", Texas, August 1986.
- [14] Nguyen, T., "Image Blurring Effects Due to Depth Discontinuities", *Technical Note ISP-1080*, University of Illinois, May 1990.

Illuminant Precompensation for Texture Discrimination using Filters

Robert S. Thau*

M.I.T. AI lab, NE43-710
Cambridge, MA, 02138

Abstract

Typically, texture-discrimination algorithms have been tested on images containing either mosaics of synthetic textures, or artificially created mosaics of real textures — in any case, images in which most of the changes in intensity can be ascribed to the textures themselves. However, real images aren't formed like this, and may contain steep gradations in intensity which have nothing to do with local texture, such as those caused by incident shadows.

A texture discrimination algorithm based on linear filters can fail in the presence of strong intensity edges generated by shadows or object boundaries, as they may easily contain an order of magnitude more energy than the gradations in intensity due to any texture in the image *per se*. In these cases, the mechanism may become responsive only to strong luminance effects, and not to texture. We have found that good performance on natural images containing texture can only be obtained from a filter-based texture detection scheme if it includes a stage which attempts to bring large intensity gradients within bounds. This may be accomplished by a preliminary nonlinear filtering step involving entirely local computation.

1 Statement of problem

Images contain regions of different textures; determining the boundaries between these regions is an interesting

*This paper describes research done within the Center for Biological Information Processing, in the Department of Brain and Cognitive Sciences, and at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. This research is sponsored by a grant from the Office of Naval Research (ONR), Cognitive and Neural Sciences Division, by the Artificial Intelligence Center of Hughes Aircraft Corporation, by the Alfred P. Sloan Foundation, by the National Science Foundation, by the Artificial Intelligence Center of Hughes Aircraft Corporation (S1-801534-2), and by the NATO Scientific Affairs Division (0403/87). Support for the A. I. Laboratory's artificial intelligence research is provided by the Advanced Research Projects Agency of the Department of Defense under Army contract DACA76-85-C-0010, and in part by ONR contract N00014-85-K-0124.

and important problem. A common approach is to look for texture discriminants which are somehow defined at every point in an image. One type of texture discriminant, to be explored here, is based on the outputs of a set of linear filters applied to a (preprocessed) image.

Now, the output of the filters themselves (or even the full-wave rectified output) is inadequate for use as a discriminant; it must be post-processed somehow. For example, the output of a vertically-oriented real Gabor filter applied to a vertical sine grating will have high peaks, but it will not be consistently high (it will have zeroes!). Therefore, changes in the filter's output do not, in and of themselves, signal changes in the underlying texture, and so they cannot be used directly to locate texture boundaries. Rather, we must process the filter outputs further, in order to obtain a discriminant which does not vary so long as the texture stays the same. This problem has received considerable attention in the literature, and there have been many proposed solutions; see variously [Malik & Perona, 1990], [Bovic et al., 1990], [Reed & Wechsler, 1990], [Turner, 1985], etc. The approach to computing discriminants from the outputs of the filters which we follow here is a radically simplified version of that used by Malik & Perona, one which, it should be noted, they specifically consider and reject, due to its inability to match human performance on certain texture pairs. (Other salient differences between this work and that of Malik & Perona are the use of a different set of filters, and of course, the absence in their model of any effective illuminant precompensation).

But there is another problem. Even if we have a computation which yields workable discriminants for artificial texture samples, or mosaics of natural texture samples, that still does not guarantee good performance on real images. The trouble is that real images contain features other than texture which may excite the filters, such as shadows or strong intensity edges. Even if these irrelevant features are not what the filters are directly tuned for, the filters will still respond if the features are sufficiently strong. For instance, if the image contains a very strong vertical intensity edge, then any filters tuned to respond to anything like a vertical edge will produce a response.

If the filters' responses to non-textural features are comparable to the responses produced by elements of the textures which are present, then a properly designed

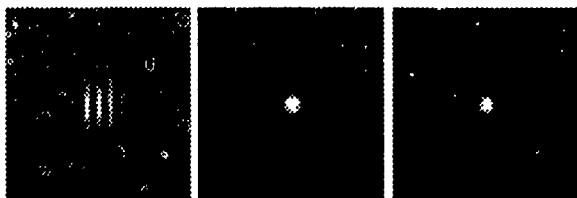


Figure 1: Point spread functions of the filters — Gabor, DOG1, and DOG2, respectively.

discriminant mechanism will simply reject them as spurious. However, if the responses produced by non-textural features are much stronger than the responses produced by actual texture, then suppressing them can become a serious problem (as shown in the figures below). One approach which we might choose to deal with these spurious features would be to run an edge detector on the image, and to suppress the texture computation in the presence of strong intensity edges. In this paper, we have chosen a different approach, namely, to preprocess the image so that the intensity edges in the textures will be as strong as any others in the image.

2 The Algorithm

We have considered two algorithms, of which only one is presented here; I mention this only to point out we have found that the exact nature of the best preconditioner may actually depend on what happens in subsequent computations.

- We begin with an image I and a collection of filters F_i .
- The image is conditioned, with the intent of reducing the influence of illumination on succeeding steps. This conditioning step is the main subject of this paper; remaining steps form the discriminant mechanism.
- The filters F_i are applied to the image I , yielding raw results R_i . These are divided into positive and negative parts, R_{i+} and R_{i-} .
- The R_{i+} and R_{i-} are further processed to yield a set of discriminants, D_j , which are still defined, pointwise, across the image, being computed from local values of R_{i+} and R_{i-} .
- A "texture gradient" is computed, as by Malik & Perona, the pointwise maximum of the magnitudes of the gradients of the discriminants, that is,

$$G = \max_i \|\nabla D_i\|$$

- Location of texture boundaries is determined by applying non-maximum suppression and adaptive thresholding to the texture gradient, as in the Canny edge detector [Canny, 1983].

The end result of this process is a binary field which contains isolated "texture edges". In the MIT vision machine, these fields could be used as input to a Markov Random Field process for assigning types to the intensity edges to the image [Geman & Geman, 1984, Gamble,

1987 Geiger et al., 1989]. Alternatively, given sufficient computational resources, one could use the individual discriminants, or various combinations of the discriminants (e.g. by scale), as input to an MRF-based integration process.

3 Image preconditioning

An ideal preconditioning step would completely discount the effects of illumination on the texture discriminants, and hence, on the detected texture boundaries. It is impossible to disentangle the two completely, of course. Consider, for instance, a movie — all the texture in the perceived images, along with everything else, results solely from the highly structured illuminant, the screen itself, on which the images are projected, is untextured and flat. I will restrict myself here to heuristics which attempt to compensate for variations in illumination which is on a scale several times larger than the scale of the texture itself (e.g., changes in global scene illumination, and shadows).

In the usual model of image formation, the light reflected by a surface patch is the reflectance of that patch multiplied by the incident light (with compensation for the difference between the angle of incidence and the angle of observation). If we assume that the angle of incidence and the intensity of the incident light will both tend to be constant over large regions of the image, the effect of the illuminant is reduced to a constant factor multiplying the image intensities within a local region, the exact value of which may vary. We seek to remove the influence of this constant factor. A simple-minded approach would be to divide the image intensity at each pixel by the average image intensity in a neighborhood centered on that pixel.

Indeed, an ad hoc method which works well on many textured images is only slightly more sophisticated than this. Rather than computing a simple average (which would be susceptible to rapid fluctuation from pixel to pixel), I smooth the image with a Gaussian filter, and divide the raw image intensity at each point by the response of the filter at that point:

$$I' = I / (S * I)$$

where S is a Gaussian smoothing filter. It is interesting to note that similar approaches to removing the influence of the illuminant have been found useful in maintaining color constancy, another problem which may be phrased in terms of removing the effects of the illuminant. (This is in some sense analogous to the prefiltering necessary to enforce color constancy in schemes such as those of [Land, 1986] and [Hurlbert & Poggio, 1988]).

Note that we have now introduced a new parameter — the choice of the standard deviation of the filter S . Perhaps surprisingly, I found that it works best to choose S to be quite small, well under the size of the scale parameters of the largest filters in use ($\sigma=5$ pixels). Larger values worked somewhat worse; see below.

4 The filters

The basis of the texture discrimination algorithm I discuss here is the application of a set of oriented and un-

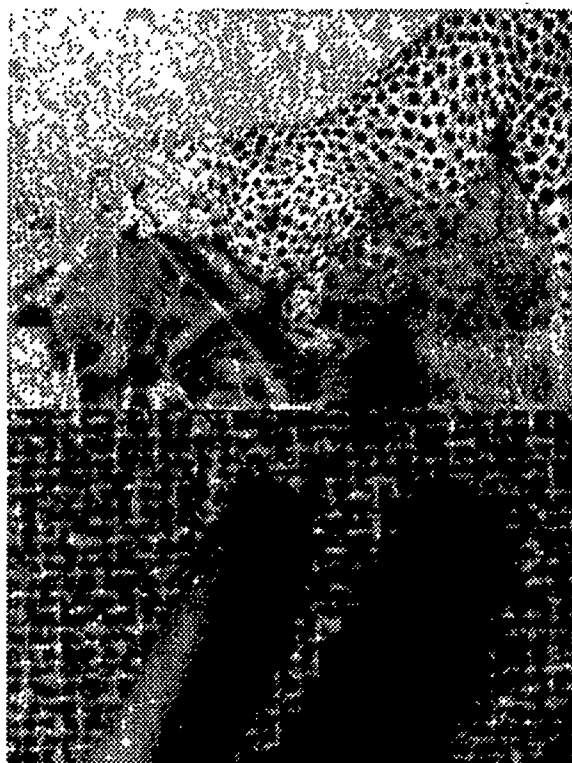


Figure 2: Results with smoothed filter half-outputs and illuminant compensation by division

oriented linear filters to the image. The oriented filters were Gabor functions of various orientations and scales. The oriented filters were sums of Gaussians, corresponding to Malik & Perona's two unoriented filters, described as being of type DOG1 and DOG2.

Note that only one phase of the Gabor filters was used. A more conventional approach would be to use a quadrature pair, and combine the results of some sort of energy measure. This was not done here for reasons of efficiency; instead, we used one element of the pair, and computed an energy-like measure by smoothing. This may not seem like much of a gain of efficiency; we have removed an oriented filter, and replaced it with a smoothing filter. However, the smoothing filter is separable (in fact, a Gaussian), and thus can be computed much more efficiently than the nonseparable orientation-tuned filter it replaces. All filters used were of even phase. (In this, we follow Malik & Perona, whose choice of even phase was dictated by the apparent absence of odd-symmetric mechanisms in human texture discrimination).

Now, in a real image, features occur at a variety of scales and orientations. The filters used by a texture discrimination algorithm must be able to pick up all these features. The brute force approach taken here (as, apparently, in the mammalian brain), is to use filters tuned to several different scales. All examples used in this paper use filters whose overall size (σ) parameter is chosen from 2, 3, 4, 6, 8 or 10 pixels. The two unoriented filters were replicated at each scale. Also present at each scale were six oriented filters, whose orientations were evenly

spaced around the circle. Point spread functions of the two sets of filters are in figure 1.

In addition to the Gabor filters, we also experimented with oriented filters produced by summing offset oriented Gaussians. In general, the type of oriented filters in use didn't affect which types of texture boundaries could be found by the algorithms. However, with the Gabor functions, peaks in the computed "texture gradients" were generally sharper and more prominent, leading to somewhat better performance overall. All figures in this paper were computed with Gabor-function oriented filters.

5 A simple discriminant — smoothing filter half-responses

The simplest, crudest, way of computing a texture discriminant is to smooth the outputs of the positive and negative parts of the filters with some appropriately chosen Gaussian. That is, for each filter, we compute two discriminants:

$$D_{i+} = M * (R_{i+})$$

and

$$D_{i-} = M * (R_{i-})$$

where R_{i+} is the positive response of the filter F_i to the preconditioned image I' , and similarly for R_{i-} . M is the Gaussian smoothing filter, for all examples in this paper, I chose the standard deviation of M to be twenty-five pixels. (Recall that the largest scale parameters for the filters in use was ten pixels, corresponding to a maximum texon size of that order. This constitutes smoothing over

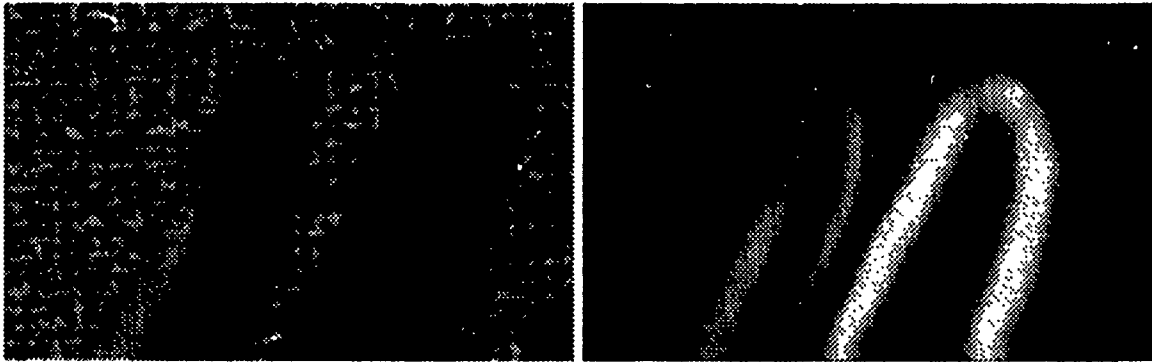


Figure 3: Results on the same images as in fig.1, with no preconditioning. Note that the mechanism's responses are determined mostly by the placement of strong intensity edges. Texture boundaries which are not accompanied by an intensity edge receive a meager response.

two or three textons diameters, for the largest textons that the filters could be expected to detect).

Applying this scheme with a set of Gabor filters and difference-of-gaussian filters at various scales, we obtain the results shown in figure 2. Note that the algorithm reliably picks up texture boundaries, and is insensitive to pure intensity edges. This is particularly noticable of the image of a finger casting a shadow on a wicker background, in which we see the boundary of the finger completely found, and the shadow (a strong intensity edge) not found. (The latter is legitimate; the textures on either side of the boundary are identical). As we shall see, other ways of attempting to compensate for areal intensity variations yield much poorer results. Note also that the algorithm succeeds in finding even fairly subtle texture boundaries, such as the boundary between the legs and the body of the cheetah, and that it detects orientation discontinuities as well as differences in texton density.

6 Alternative preconditioners

In this section, I compare performance the preconditioner described above with several alternatives (including having no preconditioner at all), which all perform worse.

6.1 No preconditioning

When adding a stage such as preconditioning to one's model, one can never help wondering whether the step is really necessary. As results in figure 3 show, a preconditioning does appear to be necessary. Without the preconditioner, performance on synthetic images is pretty much unaffected, but as the figure shows, performance on real images is substantially worse.

6.2 Variants on the division preconditioner

As mentioned above, the choice of the mask in the smooth-and-divide image preconditioner is essentially arbitrary. Figure 4 illustrates results for one image, in which the mask is replaced by Gaussian masks with different space constants. Note that as the size of the areas being averaged increases, the algorithm becomes less

likely to find the true texture discontinuity, and more likely to pick up the shadow, which is not a texture discontinuity. (In the limit of extremely large areas, we would find ourselves dividing each pixel value by the average intensity of the image, which is, of course, equivalent to no compensation at all).

6.3 Pointwise logarithm

The observation behind this approach is that if the observed image intensity I is the product of the reflectance R and some undetermined (and undesirable) luminance factor L , then taking logs turns $I = RL$ into $\log I = \log R + \log L$, turning the multiplicative factor into an additive one, which should have less of an influence on the output of the linear filters. Unfortunately, a strong edge in the intensities in an image is also present in their logarithms. As a result, not only do we find the "texture detector" picking up the intensity boundaries at the shadow, but it also picks up an intensity boundary within the homogeneously textured finger itself. Results are shown in fig. 4.

6.4 Normalizing filter values

The rationale here is that a constant factor affects the output of all filters equally, so normalizing the output of all filters to keep their sum a constant should cancel out the constant-factor effects of the illuminant. The approach suffers a fatal flaw: in large, featureless regions of constant intensity where no filter has a substantial output, the minor noise in the filters' outputs gets amplified out of all proportion to reality.

7 Conclusion

This paper has explored a simple, but effective, texture discrimination algorithms based on the application of linear filters to images.

It has shown that in order to get good performance on natural images (especially those containing shadows), it is necessary to preprocess the images to remove strong intensity gradients. In the absence of such preprocessing, differences in intensity from region to region in the image can "hijack" the texture discriminator. In many natural

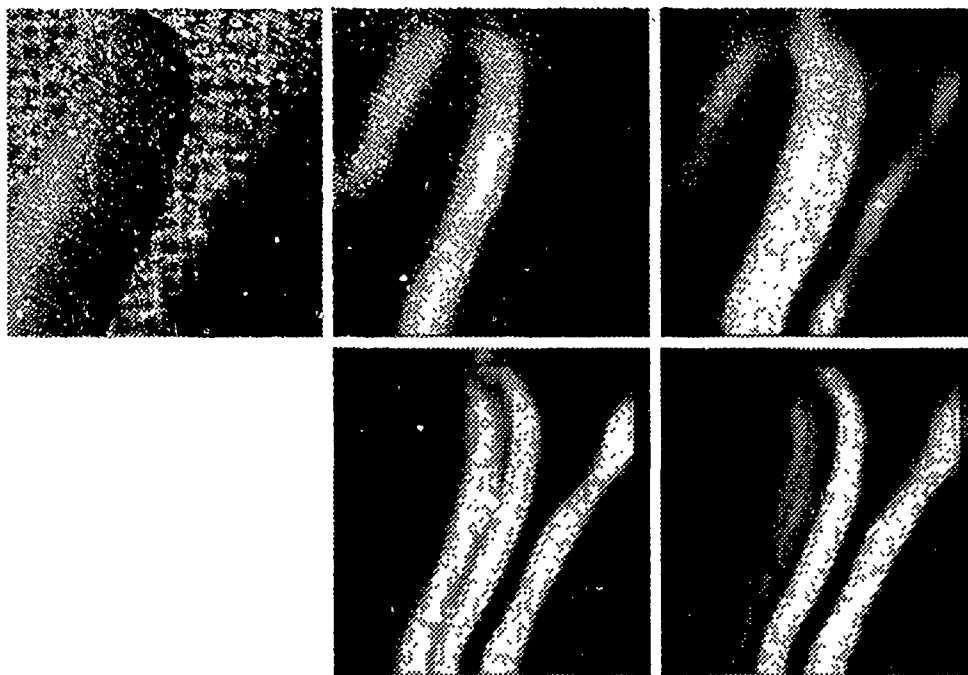


Figure 4: Texture gradients for the same image, with different scales on the smoothing for illuminant compensation. In order of decreasing quality: $\sigma = 5$ pixels (the default), ten, fifteen, or twenty.

images broad intensity differences may be detected as texture edges, even if the texture on both sides of the discontinuity is the same (as for shadows), while texture discontinuity *without* an intensity difference may yield only a weak signal, which could easily be neglected. Most texture detectors in the recent literature have not had such a mechanism. Indeed, a texture discriminator based on, say, blob detection, is able to live without one, as only the few blobs on the boundary are substantially disturbed. For a texture discriminator based on linear filters, however, such a mechanism is key.

8 Acknowledgments

Tomaso Poggio first suggested I work on filter-based approaches to texture discrimination, in 1988; I thank him, Pietro Perona, Shimon Edelman, Brian Subirana, Ed Gamble, Rick Lathrop, Harry Voorhees, and Charles Chubb, among others, for their discussions and comments, and Ed, Rick, and Terry Webster for moral support.

References

- [1] Bergen and Adelson, "Early Vision and Texture Perception", *Nature*, 333, pp. 363-4
- [2] Bovik, Clark and Geisler, "Multichannel Texture Analysis Using Localized Spatial Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, p. 56. 1990
- [3] Canny, J., *Finding Edges and Lines in Images*, Technical report 720. MIT AI laboratory. 1983
- [4] Gamble, E., and Poggio, T., "Visual Integration and Detection of Discontinuities: The Key Role of Intensity Edges", MIT AI lab memo 970, 1987
- [5] Geiger, D., and Girosi, F., "Parallel and Deterministic Algorithms for MRFs: Surface Reconstruction and Integration", MIT AI lab memo 1114, 1989
- [6] Geman and Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721-741, 1984
- [7] Gurnsey, R. and Browse, R., "Micropattern Properties and Presentation Conditions Influencing Visual Texture Discrimination," *Perception and Psychophysics*, 41, pp. 239-252, 1987
- [8] Grossberg, S. and Mingolla, E., "Neural Dynamics of Perceptual Grouping: Textures, Boundaries, and Emergent Segmentations", *Perception and Psychophysics*, vol. 38, pp. 141-171, 1985
- [9] Hurlbert, A., and Poggio, T., "Synthesizing a Color Algorithm from Examples", *Science*, Vol. 239, 482-485, 1988
- [10] Hurlbert, A., and Poggio, T., *Learning a Color Algorithm from Examples*, M.I.T. AI lab memo 909, 1987
- [11] Malik, J., & Perona, P., Preattentive Texture Discrimination with Early Vision Mechanisms, *J. Opt. Soc. Am. A*, Vol. 7, pp. 923-33
- [12] Julesz, B., "Textons. the Elements of Texture Perception and their Interactions," *Nature*, Vol. 290, pp. 91-97, 1981a

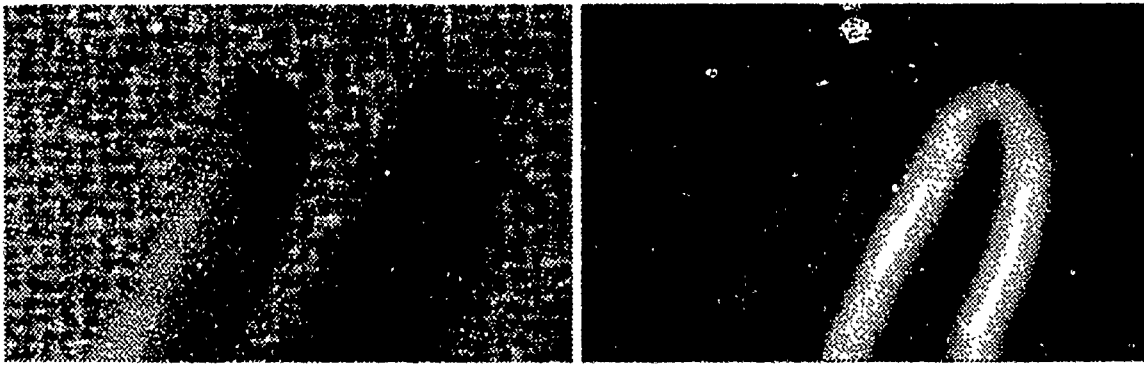


Figure 5: Results on same image as in fig.2, with pointwise logarithm

- [13] Julesz, B., "A Theory of Preattentive Texture Discrimination Based on First-Order Statistics of Textons," *Biological Cybernetics*, Vol. 41, pp. 131-138, 1981b
- [14] Julesz, B., "Texton Gradients: The Texton Theory Revisited," *Biological Cybernetics*, Vol. 54, pp. 245-251, 1986
- [15] Julesz, B., and Bergen, J.R., "Textons, The Fundamental Elements in Preattentive Vision and Perception of Textures," *The Bell System Technical Journal*, Vol. 62, pp. 119-145, 1983
- [16] Land, E., "An Alternative Technique for the Computation of the Designator in the Retinex Theory of Color Vision", *Proc. Nat. Acad. Sci.*, Vol. 83, pp. 3078-3080, 1986
- [17] Riley, M., *The Representation of Image Texture*, MIT AI lab Technical Report 649, 1981
- [18] Reed and Wechsler, "Segmentation of Textured Images and Gestalt Organization Using Spatial/Spatial-Frequency Representations", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, p. 1, 1990
- [19] Turner, M., "Texture Discrimination by Gabor Functions", *Biological Cybernetics*, Vol. 55, pp. 71-82, 1986
- [20] Van Gool, L., Dewaele, P., and Oosterlinck, A., "Texture Analysis Anno 1983", *Computer Vision, Graphics, and Image Processing*, vol 29, pp. 336-357, 1985
- [21] Voorhees, H., *Finding Texture Boundaries in Images*, Technical report 986, MIT Ai laboratory, 1987
- [22] Voorhees, H., & Poggio, T., Computing Texture Boundaries from Images, *Nature*, 333:364-367, 1988

Surface Reflection: Physical and Geometrical Perspectives

Shree K. Nayar, Katsushi Ikeuchi, and Takeo Kanade

The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213

Abstract

Machine vision can greatly benefit from the development of accurate reflectance models. There are two approaches to the study of reflection: physical and geometrical optics. While geometrical models may be construed as mere approximations to physical models, they possess simpler mathematical forms that often render them more usable than physical models. However, geometrical models are applicable only when the wavelength of incident light is small compared to the dimensions of the surface imperfections. Therefore, it is incorrect to use these models to interpret or predict reflections from smooth surfaces, and only physical models are capable of describing the underlying reflection mechanism.

This paper is directed towards unifying physical and geometrical approaches to describe reflection from surfaces that may vary from smooth to rough. More specifically, we consider the Beckmann-Spizzichino (physical optics) model and the Torrance-Sparrow (geometrical optics) model. We have chosen these two models in particular as they have been reported to fit experimental data very well. Each model is described in detail, and the conditions that determine the validity of the model are clearly stated. From studying the behaviors of both models, we propose a model comprising three reflection components: the diffuse lobe, the specular lobe, and the specular spike. The dependencies of the three components on the surface roughness and the angles of incidence and reflection are analyzed in detail.

1 Introduction

Most machine vision problems involve the analysis of images resulting from the reflection of light. The apparent brightness of a point depends on its ability to reflect incident light in the direction of the sensor: what is commonly known as its reflectance properties. Therefore, the prediction or interpretation of image intensities requires a sound understanding of the various mechanisms involved in the

reflection process. While shape extraction and object recognition methods are being refined, it is also essential for the vision community to research and utilize more sophisticated reflectance models. Once a "general" reflectance model is made available, we are free to make reflectance assumptions that are reasonable for the vision application at hand. The resulting more specific model may then be used to develop efficient perception techniques.

Various reflectance models have been used in the areas of machine vision and graphics. Horn [8] used the Lambertian diffuse reflectance model and the double-delta specular reflectance model to develop shape-from-shading algorithms for machine vision. Horn [7] has also provided an excellent review of some of the early models used in graphics for hill shading. Phong [20] proposed a parametrized continuous function to represent specular reflectance, and used the model to produce computer-synthesized images of objects. Woodham [33] used the Lambertian model to determine object shape by means of photometric stereo. Ikeuchi [12] used the double-delta specular model to determine the shape of specular surfaces by photometric stereo. Pentland [19] developed a local shape-from-shading algorithm that assumes Lambertian reflectance. Coleman and Jain [4] proposed the four-source photometric stereo, which discards specular reflections and uses the diffuse reflections and the Lambertian model to determine shape information. Sanderson, Weiss, and Nayar [25] have used the double-delta specular model to determine the shape of specular surfaces by means of the structured highlight technique. Recently, Nayar, Ikeuchi, and Kanade [16] have developed the photometric sampling method that uses a hybrid reflectance model, comprised of both Lambertian and specular models, to extract the shape and reflectance of Lambertian, specular, and hybrid surfaces.

The above applications have proven that the Lambertian model does reasonably well in describing diffuse reflections. Moreover, its simple functional form has made it a popular reflectance model in the vision research community. On the other hand, the specular models used above perform well only when the object surface is very smooth, in which case, most of the reflected light is concentrated around the

specular direction. Specular reflection from rough surfaces, however, requires careful examination, and its dependence on the imaging and illumination geometry can only be obtained by a formal treatment of optics. There are two different approaches to optics, and thus two different approaches to the study of reflection. The physical optics approach uses electromagnetic wave theory to study the reflection of incident light. The geometrical optics approach, on the other hand, uses the short wavelength of light to simplify the reflection problem. Hence, geometrical models may be viewed as approximations to physical models.

The Beckmann-Spizzichino physical optics model and the Torrance-Sparrow geometrical optics model have recently attracted considerable attention. Both models have been developed to describe specular reflection mechanisms, and both have been found to fit experimental data quite well [11] [31]. Owing to its simpler mathematical form, the Torrance-Sparrow model is more popular than the Beckmann-Spizzichino model, and has been used in the areas of computer vision and graphics. Healey and Binford [6] have used the Torrance-Sparrow model to determine local shape from specular reflections. Wolff [32] has used the model to develop spectral and polarization stereo methods. Cook and Torrance [5] have modified the model and used it to render images of objects. Tagare and Figueiredo [30] have discussed both the Beckmann-Spizzichino and the Torrance-Sparrow models in their survey of various reflection mechanisms.

When applying physical and geometrical models, it is important to satisfy the conditions that determine the validity of the models. This requires an understanding of the restrictions imposed by the assumptions made while developing the models. Most of these assumptions are related to the microscopic shape and physical properties of the reflecting surface. In this paper, we study these assumptions in detail and determine which reflection mechanisms are described only by physical models and which ones may be approximated by the relatively simpler geometrical models. Our objective, therefore, is to unify physical and geometrical reflectance models to develop a single model that can describe reflection from surfaces that may vary from smooth to rough. In the process of achieving this goal, we address the following questions:

- How is the microscopic shape of a surface modeled, and when is a surface rough?
- How are physical optics and geometrical optics models developed?
- Under what conditions are the Beckmann-Spizzichino and the Torrance-Sparrow models valid?

- How do the reflectance curves predicted by these two models compare with one another, and how are the surface roughness parameters of the two models related to each other?
- What are the primary components of surface reflection, and which model should be used to represent each of the primary components?
- How are the reflection components dependent on the surface roughness, and on the angles of incidence and reflection?

The paper is structured as follows. In section 2, we define radiometric concepts that are useful in the analysis of surface reflection. In section 3, we look at different approaches to modeling surface profiles. In section 4, we highlight the main steps that are involved in the derivation of the Beckmann-Spizzichino and Torrance-Sparrow models, and clearly state the assumptions made in the process of their development. On the basis of the reflectance curves predicted by the two models, we propose a reflectance model that has three primary components: the *diffuse lobe*, the *specular lobe*, and the *specular spike*. In section 5, we study these reflectance components in detail.

2 Radiometric Definitions

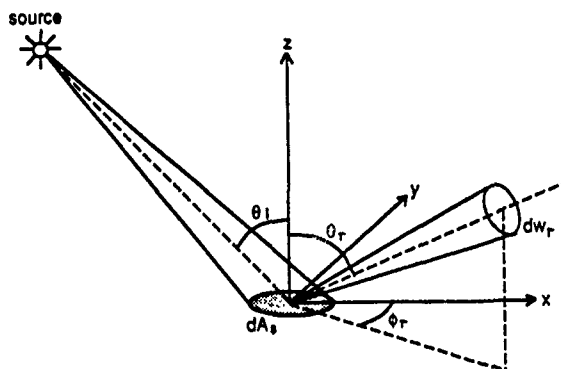


Figure 1: Basic geometry needed to define radiometric terms.

In this section, we present definitions of radiometric terms that are useful in the study of surface reflection. Detailed derivations and descriptions of these terms are given by Nicodemus et al. [18]. As shown in Figure 1, all directions are represented by the zenith angle θ and the azimuth angle ϕ . The light source is assumed to lie in the x - z plane and is therefore uniquely determined by its zenith angle θ_i . The monochromatic flux $d\Phi_i$ is incident on the surface area dA_s from the direction θ_i , and a fraction of it, $d^2\Phi_r$, is reflected

in the direction (θ_r, ϕ_r) . The irradiance I_s ¹ of the surface is defined as the incident flux density:

$$I_s = \frac{d\Phi_i}{dA_s} \quad (1)$$

The radiance L_r of the surface is defined as the flux emitted per unit fore-shortened area per unit solid angle. The surface radiance in the direction (θ_r, ϕ_r) is defined as:

$$L_r = \frac{d^2\Phi_r}{dA_s \cos\theta_r d\omega_r} \quad (2)$$

The BRDF (Bi-Directional Reflectance Distribution Function) f_r of a surface is a measure of how bright the surface appears when viewed from a given direction, when it is illuminated from another given direction. The BRDF is defined as:

$$f_r = \frac{L_r}{I_s} \quad (3)$$

In the following sections of this paper, we will frequently use the above radiometric definitions.

3 Surface Model

The manner in which light is reflected by a surface is dependent on, among other factors, the microscopic shape characteristics of the surface. A smooth surface, for instance, may reflect incident light in a single direction, while a rough surface will tend to scatter light in various directions, maybe more in some directions than others. To be able to accurately predict the reflection of incident light, we must have prior knowledge of the microscopic surface irregularities; in other words, we need a model of the surface. All possible surface models may be divided into two broad categories: surfaces with exactly known profiles and surfaces with random irregularities. An exact profile may be determined by measuring the height at each point on the surface by means of a sensor such as the *stylus profilometer*. This method, however, is quite cumbersome and also inapplicable in many practical situations. Hence, it is often convenient to model a surface as a random process, where it is described by a statistical distribution of either its height above a certain mean level, or its slope with respect to its mean (macroscopic) slope. In this section, we discuss these two approaches to surface modeling in greater detail and explain how surface roughness is pertinent to the study of reflection.

3.1 Height Distribution Model

¹Irradiance is usually [8] denoted by the symbol E . In the following sections, we will be using E to denote the electric field, and therefore we will denote irradiance by I to avoid confusion.

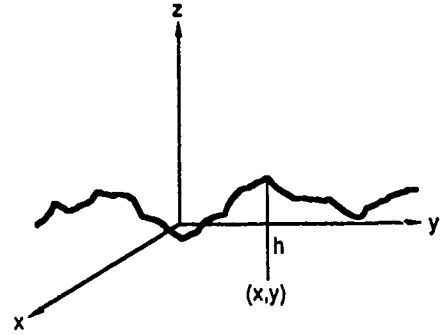


Figure 2: Surface height as a random function of the spatial coordinates.

The height coordinate h of the surface may be expressed as a random function of the coordinates x and y , as shown in Figure 2. The shape of the surface is then determined by the probability distribution of h . For instance, let h be normally distributed, with mean value $\langle h \rangle = 0$, and standard deviation σ_h . Then, the distribution of h is given by:

$$p_h(h) = \frac{1}{\sqrt{2\pi}\sigma_h} e^{-\frac{h^2}{2\sigma_h^2}} \quad (4)$$

The standard deviation σ_h is also the root-mean-square of h and represents the *roughness* of the surface. The surface is not uniquely described by the statistical distribution of h , however, as it does not tell us anything about the distances between the hills and valleys of the surface. In Figure 3, both surfaces (a) and (b) have the same height distribution function, i.e. the same mean value and standard deviation. In appearance, however, the two surfaces do not strongly resemble each other. In order to strengthen our surface model, we use an autocorrelation coefficient $C(\tau)$ that determines the correlation (or lack of independence) between the random values assumed by the height h at two surface points (x_1, y_1) and (x_2, y_2) , separated by a distance τ . We describe the autocorrelation coefficient by the fairly general function:

$$C(\tau) = e^{-\frac{\tau^2}{T^2}} \quad (5)$$

where T is the *correlation distance*, for which $C(\tau)$ drops to the value e^{-1} . We see that the surfaces (a) and (b) shown in Figure 3 have small and large correlation distances, respectively. By varying the parameters σ_h and T of our surface model, we can generate surfaces that match in appearance almost any rough surface met in practice. Moreover, if we are dissatisfied with the performance of the model, we can always use another height distribution function and/or another autocorrelation function than the ones given above.

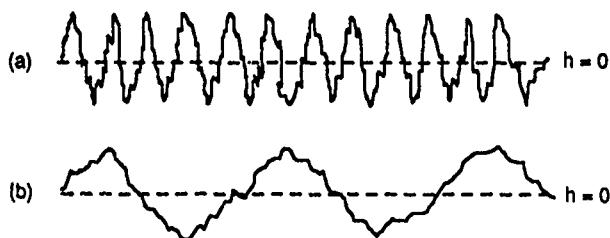


Figure 3: Random surfaces with (a) small, (b) large correlation distances.

3.2 Slope Distribution Model

It is sometimes convenient to think of a surface as a collection of planar micro-facets, as illustrated in Figure 4. A large set of micro-facets constitutes an infinitesimal surface patch that has a mean surface orientation \mathbf{n} . Each micro-facet, however, has its own orientation, which may deviate from the mean surface orientation by an angle α . We will use

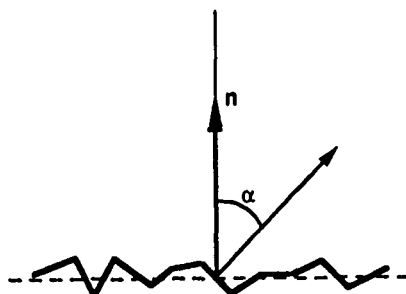


Figure 4: Surface modeled as a collection of planar micro-facets.

the parameter α to represent the slope of individual facets. Surfaces can be modeled by a statistical distribution of the micro-facet slopes. If the surface is isotropic, the probability distribution of the micro-facet slopes can be assumed to be rotationally symmetric with respect to the mean surface normal \mathbf{n} . Therefore, facet slopes may be described by a one-dimensional probability distribution function. For example, the surface may be modeled by assuming a normal distribution for the facet slope α , with mean value $\langle \alpha \rangle = 0$ and standard deviation σ_α :

$$p_\alpha(\alpha) = \frac{1}{\sqrt{2\pi}\sigma_\alpha} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}}. \quad (6)$$

The surface model in this case is determined by a single parameter, namely, σ_α , unlike the height distribu-

tion model, which requires two parameters. Larger values of σ_α may be used to model rougher surfaces. While the importance of an autocorrelation coefficient was shown for the height model, the concept of slope correlation is more difficult to interpret and, therefore, is not of much use in the generation of surfaces. The advantages of using a single parameter come with the cost of a weaker model when compared to the height model. Given a probability distribution function for α , it is difficult to visualize the shape of the surface and to estimate the root-mean-square height of the surface. However, the slope distribution model is popular in the analysis of surface reflection, as the scattering of light rays has been found to be dependent on the local slope of the surface and not the local height of the surface. For this reason, the slope model, though relatively ambiguous, is more directly applicable to the problem of surface reflection. Shortly, we will see how both height and slope models are used to develop surface reflection models.

3.3 What is a Rough Surface?

One would expect humans to respond to this question with a variety of answers. We seem to have a rather loose definition of the term "roughness." A surface that appears to be rough from a short distance may appear to be smooth from far away. In some cases, by changing the direction of illumination, surface imperfections can be made less visible and a rough surface can be made to appear smooth. If the observer is unable to discern from its appearance how rough the surface is, he or she is inclined to feel the surface and make a judgment on the basis of the resulting sensation.

In contrast to the human definition of roughness, surface reflection theories offer a stronger definition: one that relates surface irregularities to the wavelength of incident light and the angle of incidence. For incident light of a given wavelength, the roughness of a surface may be estimated by studying the manner in which the surface scatters light in different directions. If the surface irregularities are small compared to the wavelength of incident light, a large fraction of the incident light will be reflected *specularly* in a single direction. On the other hand, if surface irregularities are large compared to the wavelength, the surface will scatter the incident light in various directions. Conversely, the same surface can be made to appear smooth or rough by varying the wavelength of incident light; or for the same wavelength it can be made to appear smooth or rough by varying the angle of incidence.

Rayleigh suggested a way of relating surface roughness to wavelength and angle of incidence, and established a simple criterion for classifying surfaces as smooth or rough. Consider rays 1 and 2 in Figure 5, incident at an angle β on a surface with irregularities of height H . Since

the two rays strike the surface at locally smooth patches, both rays are specularly reflected. The rays originate from a source plane that is perpendicular to the rays, and they are received by a detector plane that is perpendicular to the reflected rays. We are interested in finding the difference

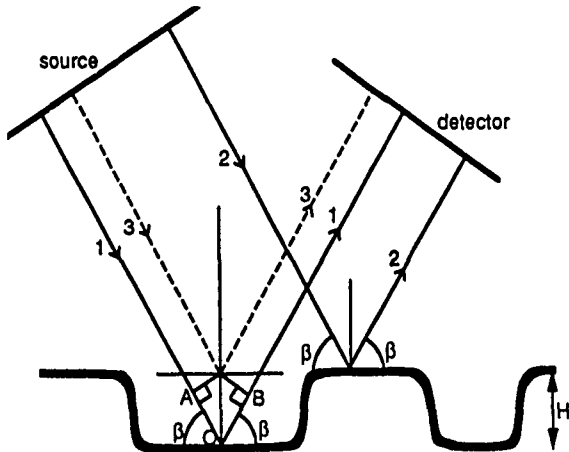


Figure 5: Analyzing surface roughness from the view point of reflection.

between the paths traveled by the two rays. Using basic geometry it can be shown that ray 2 and the imaginary ray 3 travel the same distance. Therefore, the path difference Δd between rays 1 and 2 is equal to the path difference AOB between rays 1 and 3, and is determined as:

$$\Delta d = 2H \sin \beta. \quad (7)$$

If λ is the wavelength of the incident rays, the phase difference between the rays received by the detector may be determined from the path difference as:

$$\Delta \Omega = \frac{4\pi H}{\lambda} \sin \beta. \quad (8)$$

When $\Delta \Omega$ is very small, the two rays received by the detector will be almost in phase with each other, and the received energy will be nearly equal to the sum of the energies carried by the two rays. In this case, the surface reflects light specularly. However, as the phase difference approaches π , the two rays will be in phase opposition and will tend to cancel the effects of each other. In fact, at $\Delta \Omega = \pi$ no energy will flow in the direction of the detector. The incident energy is thus redistributed in other directions, and the law of conservation of energy is preserved. Hence, the extreme cases are: $\Delta \Omega = 0$, when the surface reflects light specularly and is thus smooth; and $\Delta \Omega = \pi$, when the surface scatters light and is rough. We can thus classify surfaces as smooth and rough by picking an arbitrary threshold between $\Delta \Omega = 0$ and $\Delta \Omega = \pi$. By selecting a threshold value of $\pi/2$ we have the *Rayleigh*

criterion that states that a surface is considered to be rough when:

$$H > \frac{\lambda}{8 \sin \beta}. \quad (9)$$

This is, of course, a rather simple approach to determining the roughness of a surface. Some papers that discuss the height distribution model have defined a rough surface as one whose root-mean-square height is much greater than the wavelength of incident light, i.e. $\sigma_h \gg \lambda$. More sophisticated criteria have been developed since the Rayleigh criterion was first proposed. We will not pursue these criteria here but direct the interested reader to [1] for a more detailed treatment. In fact, we have described the Rayleigh criterion only to bring forth the concept of roughness and to emphasize its significance in the study of surface reflection.

4 Reflection Model

When light is incident on a boundary interface between two different media, it is reflected according to well-known laws. There are two different approaches to optics and, consequently, two different approaches to the study of reflection. *Physical* or *wave* optics is based directly on electromagnetic wave theory and uses Maxwell's equations to study the propagation of light. *Geometrical* or *ray* optics, on the other hand, uses the short wavelength of light to simplify many of the light propagation problems. Geometrical optics is generally able to explain the gross behavior of light when the wavelength is small compared to the pertinent physical dimensions of the system (in our case, the surface imperfections).

In this section, we study surface reflection from the perspective of physical and geometrical optics. More specifically, we discuss a physical optics reflection model, namely, the Beckmann-Spizzichino model, and a geometrical optics reflection model, namely, the Torrance-Sparrow model. We highlight the main steps that are involved in the derivation of both models and clearly state the assumptions made in the process of their development. The derivations will draw on the surface modeling approaches discussed in the previous section. Later, the two models are compared by plotting the predicted reflectance as functions of viewer and source directions.

4.1 Physical Optics Model

Light is an electromagnetic phenomenon. Therefore, in a strict sense, optics should be studied as a branch of electrodynamics. Optics is usually treated as a separate field because it was studied long before its electromagnetic character was realized. Before we address the *scattering* of incident light waves by smooth and rough surfaces, we feel

that a very brief introduction to electromagnetic waves is in order.

4.1.1 Electromagnetic Waves

In the atomic theory of matter, electromagnetic effects are considered to arise from the forces exerted on each other by elementary charged particles. The elementary positive and negative particles are the proton and electron, respectively. Consider two charged particles placed in the vicinity of each other. Due to their respective charges, the particles will exert a force on each other. If the particles are at rest, they will experience a constant electrostatic force resulting from the *electric field* generated by them. However, if the particles have different relative velocities with respect to a common frame of reference, the force acting between them will differ from the electrostatic force. This statement can be verified by simple experiments [2]. The discrepancy between the forces experienced when the particles are at rest and when they are in relative motion suggests the presence of another field, namely, the *magnetic field*, in addition to the electric field. In fact, Maxwell's equations may be interpreted as a mathematical formalization of the following physical phenomenon: associated with a time-varying electric field is a magnetic field. Therefore, the forces experienced by a moving charge can be conveniently represented by means of electromagnetic field vectors: the *electric field intensity* \mathbf{E} and the *magnetic field intensity* \mathbf{H} . Conversely, an electromagnetic field may be generated by applying forces and physically moving charges in some region of space. The electromagnetic field does not require a medium for its existence. Therefore, electromagnetic energy can be radiated from the space in which the charged particles are moving, to form a traveling *electromagnetic wave*. The field equations for the electromagnetic wave can be derived directly from Maxwell's equations.

Consider the light waves radiated by a point source of light. When the source is at a large distance from the point of observation, the *spherical waves* radiated by the source may be assumed to be *plane waves*, like the one shown in Figure 6. The electric and magnetic field vectors of the plane wave may be expressed as follows:

$$\begin{aligned}\mathbf{E} &= E_0 \mathbf{e} e^{-i\mathbf{k} \cdot \mathbf{r}} e^{i\omega t} \\ \mathbf{H} &= H_0 \mathbf{h} e^{-i\mathbf{k} \cdot \mathbf{r}} e^{i\omega t}\end{aligned}\quad (10)$$

where \mathbf{k} is the wave propagation vector, \mathbf{r} is the displacement vector that determines the observation point in space, the unit vectors \mathbf{e} and \mathbf{h} correspond to the directions of the electric and magnetic fields, respectively, and the complex coefficients E_0 and H_0 represent the strengths of the electric and magnetic fields, respectively. It is important to note that, in general, the above expressions give \mathbf{E} and \mathbf{H} complex

values. However, the actual field is determined only by the real components of the field vectors, i.e. $\text{Re}[\mathbf{E}]$ and $\text{Re}[\mathbf{H}]$, and the complex notation is used only for ease of mathematical manipulation. Bearing this point in mind, we will continue to use the complex forms of \mathbf{E} and \mathbf{H} .

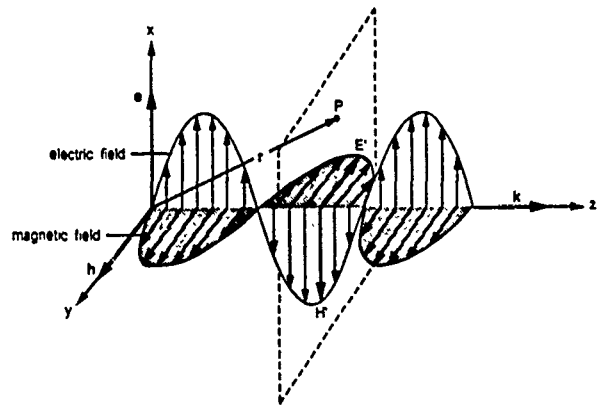


Figure 6: A plane electromagnetic wave.

The first exponential term in the above field equations suggests that the magnitudes of electric and magnetic fields vary sinusoidally as a function of the distance along the direction of propagation. The direction of the vector \mathbf{k} corresponds to the direction of propagation of the wave, while its magnitude k , called the *propagation constant*, determines the spatial frequency of the wave. The propagation constant is related to the wavelength λ of the plane wave as follows:

$$k = \frac{2\pi}{\lambda}. \quad (11)$$

If the wavelength lies between 400 nano-meters and 700 nano-meters, the wave can be detected by the human eye and is called *monochromatic light*.

The second exponential term in the field equations indicates that the field intensities also vary sinusoidally as a function of time at a radian frequency of oscillation, ω . The functions that describe the spatial and temporal field variations are dependent on the function that represents the forces applied to the charged particles to generate the wave. In most engineering applications dealing with plane waves, the field is considered to be sinusoidal steady state. Using Maxwell's equations, it can be shown that the unit vectors \mathbf{e} and \mathbf{h} are orthogonal to each other and both these vectors are orthogonal to the propagation vector \mathbf{k} . The direction of either \mathbf{e} or \mathbf{h} determines the *polarization* of the plane wave. In Figure 6, the plane wave is shown at a particular instant in time. At that instant, all points on the plane P experience the same electric and magnetic field intensities, namely, E' and H' , respectively. Therefore, the plane wave can be thought of as being constituted of infinitely large

"equi-field" planes, where each plane is perpendicular to the propagation direction \mathbf{k} .

Since time variations in the electric field are the cause of the magnetic field, and vice-versa, the amplitudes E_o and H_o of the two fields are dependent on each other, and are related as follows:

$$H_o = \sqrt{\frac{\epsilon}{\mu}} E_o, \quad (12)$$

where ϵ and μ are the *permittivity* and *permeability* of the medium, respectively. The coefficient $\sqrt{\epsilon/\mu}$ is often referred to as the *wave impedance* of the medium. Due to the above stated dependencies between the electric and magnetic field vectors, we see that an electromagnetic wave is completely defined by *either* of the two field vectors, \mathbf{E} or \mathbf{H} .

While studying surface reflection, we will be interested in determining the energy of light reflected by the surface in various directions. However, as we will see shortly, reflection models based on physical optics estimate the electromagnetic field scattered by the surface rather than the energy. Therefore, a relationship between the field and the energy carried by an electromagnetic wave would be useful. The rate of flow of *complex energy* per unit area in an electromagnetic wave can be described by a vector \mathbf{S} called the *complex Poynting vector* [2]. \mathbf{S} is defined as:

$$\mathbf{S} = \mathbf{E} \times \mathbf{H}^*, \quad (13)$$

and the quantity

$$S_a = \text{Re}[\mathbf{S}] = \frac{1}{2} \text{Re}[\mathbf{E} \times \mathbf{H}^*] \quad (14)$$

defines the time-averaged rate of flow of *physical energy* per unit area and has the dimensions watts/meter². Let E , H , and S_a be the scalar values of the \mathbf{E} , \mathbf{H} , and \mathbf{S}_a , respectively. Then the average rate of flow of energy per unit area is determined from equations 14 and 12 as:

$$S_a = \frac{1}{2} \sqrt{\frac{\mu}{\epsilon}} E E^* = \frac{1}{2} \sqrt{\frac{\epsilon}{\mu}} H H^*. \quad (15)$$

This equation will be used later to find the radiance of a surface from the electromagnetic field scattered by the surface.

4.1.2 Beckmann-Spizzichino Model

The Beckmann-Spizzichino model uses physical optics to describe the reflection of plane waves from smooth and rough surfaces. Owing to the electromagnetic character of light, this model is directly applicable to the reflection of light by surfaces. A detailed derivation of this model can be found in [1]. Our intention is to highlight the key steps

involved in the derivation of the model and to clearly state the assumptions made during its development. Later, we will study the reflectance curves predicted by the model for surfaces of differing roughness.

Consider a plane wave incident on a surface, as shown in Figure 7. All vectors and surface points are defined using the Cartesian coordinates x, y, z with origin O and unit vectors \mathbf{x} , \mathbf{y} , and \mathbf{z} . The height of the surface is determined by the function $h = h(x, y)$, and the mean level of the surface is the plane $z = 0$. The location of a surface point Q is described by its displacement vector \mathbf{r} :

$$\mathbf{r} = x\mathbf{x} + y\mathbf{y} + h(x, y)\mathbf{z}. \quad (16)$$

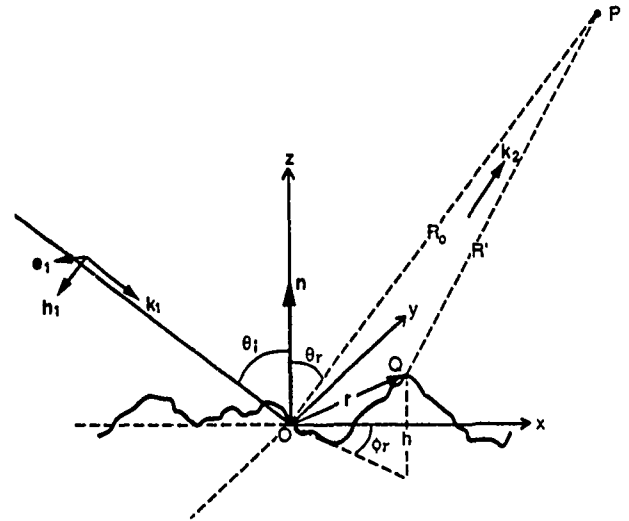


Figure 7: A plane wave incident on a rough surface, scattered in various directions.

All quantities associated with the incident field will be denoted by the subscript 1 and all those associated with the scattered field by the subscript 2. We will represent the plane wave by its electric field intensity only, keeping in mind that the magnetic field intensity may be determined from the electric field. The incident field at the surface point Q may be written as:

$$\mathbf{E}_1 = E_{o1} \mathbf{e}_1 e^{-ik_1 \cdot \mathbf{r}} e^{i\omega t} \quad (17)$$

where E_{o1} represents the electric field amplitude, \mathbf{e}_1 is the direction of the electric field, \mathbf{k}_1 is the wave propagation vector, and ω is the radian frequency of field oscillation.

We are interested in the instantaneous scattering of the incident plane wave by the surface. Hence, we can drop the second exponential term in the above equation, which

represents the temporal variation of the incident field. The incident propagation vector \mathbf{k}_1 will be assumed to always lie in the x - z plane of the coordinate frame. The angle of incidence θ , of the plane wave is the angle between the propagation vector \mathbf{k}_1 and the z axis of our coordinate frame. If we are interested in the field scattered by the surface in the direction \mathbf{k}_2 , the corresponding scattering angle θ_s is the angle between \mathbf{k}_2 and the z axis. For scattering directions that lie outside the *plane of incidence* (\mathbf{k}_1, z), we must introduce an additional angle ϕ_s , as shown in Figure 7. The propagation constant k corresponding to the propagation vectors \mathbf{k}_1 and \mathbf{k}_2 is related to the wavelength λ of the incident wave by equation 11.

The polarization of the incident wave is determined by the direction of the vector \mathbf{e}_1 . For parallel polarization, \mathbf{e}_1 lies in the plane of incidence; for perpendicular polarization, \mathbf{e}_1 is normal to the plane of incidence. An unpolarized incident wave is one whose \mathbf{e}_1 vector is neither parallel nor perpendicular to the plane of incidence, and in general, can vary in direction as a function of time. We will see later how the polarization of the incident field \mathbf{E}_1 affects the intensity of the scattered field \mathbf{E}_2 . We will not, however, concern ourselves with the polarization of the scattered field \mathbf{E}_2 , as we are only interested in the intensity of \mathbf{E}_2 . From here on, we will assume the polarization of the incident wave to be either parallel or perpendicular, and the incident field will be denoted by the scalar E_1 , where:

$$\mathbf{E}_1 = \mathbf{e}_1 E_1. \quad (18)$$

What happens when the incident plane wave strikes the surface? A simplistic description of the physical situation is as follows. A conducting surface will have an abundance of electrons that are very loosely bound to their atoms. When these electrons are subjected to the electromagnetic field carried by the incident wave, they experience forces. These forces result in a movement of the electrons, often referred to as surface currents. The surface currents give rise to new electromagnetic fields that interact with the incident field to determine the resultant field at the surface. Mathematically, the resultant field $(E)_S$ at a surface point Q must satisfy the *wave equation*²:

$$\Delta^2(E)_S + k^2(E)_S = 0, \quad (19)$$

where k is once again the propagation constant. Therefore, the field $(E)_S$ at the surface may be determined by solving the wave equation for the boundary conditions imposed by the surface profile.

The field scattered by the surface in any direction can be determined from the field at the surface. Let P be

the point of observation, and let the variable R' denote the distance between P and points on the surface S , as shown in Figure 7. We would like to find the scattered field E_2 at the point P . To this end, let us consider a volume V that is bounded almost everywhere by the surface S but is extended such that the point P lies just outside the volume. Then, it is reasonable to assume that the field $(E)_S$ is continuous, and the above wave equation must therefore be satisfied everywhere inside V . Furthermore, the point inside the volume that is nearest to P will experience almost the same field as the point P . Using these assumptions and Green's first and second theorems, the scattered field E_2 at the point P can be determined [1] from equation 19 as:

$$E_2(P) = \frac{1}{4\pi} \iint \left((E)_S \frac{\partial \psi}{\partial n} - \psi \left(\frac{\partial E}{\partial n} \right)_S \right) dS, \quad (20)$$

where:

$$\psi = \frac{e^{ikR'}}{R'}. \quad (21)$$

This is called the *Helmholtz integral*, which gives us the solution of the wave equation at any point inside (P is almost inside) a region in terms of the values of the function (surface field $(E)_S$) and its normal derivative on the boundary (the surface S) of the region. A detailed derivation of the Helmholtz integral is provided in [1]. Though it is derived for a closed surface, it is also applicable to open surfaces like the one in Figure 7.

In order to evaluate the above integral, we must find $(E)_S$ and $(\partial E / \partial n)_S$, i.e. the field and its normal derivative on the surface S . In general, these two quantities are unknown. *Kirchoff's assumption* may be used to approximate the values of the field and its normal derivative at each point on the surface. The approximation is obtained by assuming that the surface does not have any sharp edges, and thus the field at a point on the surface is equal to the field that would be present on a tangent plane at that point. Under this assumption, the field on S may be determined as:

$$(E)_S = (1 + F) E_1. \quad (22)$$

And, by differentiating this equation, the normal derivative of the field is determined as:

$$\left(\frac{\partial E}{\partial n} \right)_S = (1 - F) E_1 \mathbf{k}_1 \cdot \mathbf{n}', \quad (23)$$

where \mathbf{n}' is the normal to the surface at the point under consideration and F is the *Fresnel reflection coefficient* for a smooth plane.

Consider a plane wave incident on a smooth surface, as shown in Figure 8. As described above, the intensity of the reflected wave is determined by the surface field $(E)_S$,

²It can be shown [2] that for a source-free region of space, Maxwell's equations reduce to the wave equation.

which in turn is dependent on the surface currents. The surface currents induced by the incident wave are determined by the angle of incidence, the polarization of the incident wave, and the electrical properties (permittivity, permeability, and conductivity) of the surface medium. A fraction of the in-

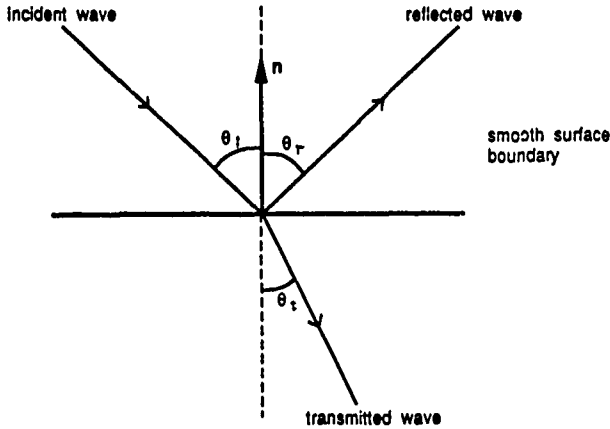


Figure 8: Light wave incident on a smooth surface.

cident electromagnetic energy, determined by these factors, will be reflected by the smooth surface, and the remaining energy transmitted by the surface. The Fresnel reflection coefficient F determines the fraction of incident energy that is reflected by the smooth surface. It is often written as $F(\theta'_i, \eta')$, where θ'_i represents the angle of incidence, and η' is the *complex index of refraction* whose value is determined by the electrical properties of the surface medium. In equations 22 and 23, F represents the fraction of the incident field that is reflected by a smooth surface. As we have shown before, the reflected energy may be determined from the reflected field by using equation 15. In deriving their reflectance model, Beckmann and Spizzichino have assumed that the incident wave is of either perpendicular or parallel polarization. The Fresnel coefficients for parallel and perpendicular polarization are, respectively [1]:

$$F_{para} = \frac{Y^2 \cos \theta'_i - \sqrt{Y^2 - \sin^2 \theta'_i}}{Y^2 \cos \theta'_i + \sqrt{Y^2 - \sin^2 \theta'_i}}, \quad (24)$$

$$F_{perp} = \frac{\cos \theta'_i - \sqrt{Y^2 - \sin^2 \theta'_i}}{\cos \theta'_i + \sqrt{Y^2 - \sin^2 \theta'_i}}. \quad (25)$$

It is important to note the difference between the angle of incidence θ_i shown in Figure 7 and the angle of incidence θ'_i in the above equations. As shown in Figure 9, the angle θ'_i is the "local" angle of incidence, i.e. the angle between the

incident wave propagation vector \mathbf{k}_i and the normal vector \mathbf{n}' at the surface point under consideration. Therefore, the angle θ'_i will have different values at different points on the surface, while θ_i is constant for a given incident wave. The term Y in the above equations is called the *normalized admittance* of the surface medium and is a function of the complex index of refraction η' . Hence, Y is also a function of the electrical properties of the medium. For a *conductor*, Y approaches infinity, while for a *dielectric* (non-conductor), Y is almost zero.

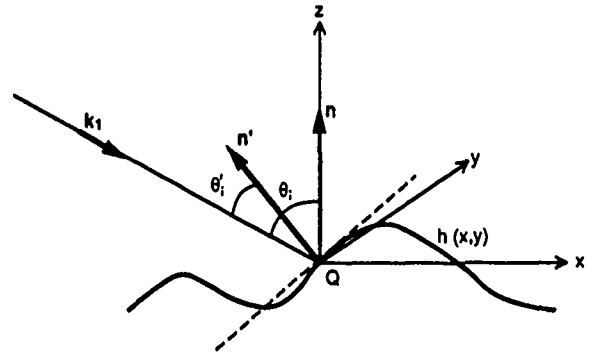


Figure 9: The "local" scattering geometry. The local angle of incidence θ'_i and the local surface orientation \mathbf{n}' may differ from the global angle of incidence θ_i and the mean surface orientation \mathbf{n} .

Let us now return to the problem of finding the scattered field E_2 by evaluating the Helmholtz integral given by equation 20. Let us assume that the surface under consideration is a rectangular patch of area A and dimensions $2X$ and $2Y$ in the x and y directions, respectively; i.e. $A = 4XY$. Further, we assume that the observation point P is at a great distance from the surface compared to the physical dimensions of the surface patch and, as a result, the vector \mathbf{k}_2 is constant over the entire surface area. Therefore, it can be seen from Figure 7 that, for any surface point, the distance R' can be expressed in terms of the distance R_o and the displacement vector \mathbf{r} as:

$$kR' = kR_o - \mathbf{k}_2 \cdot \mathbf{r}. \quad (26)$$

By substituting equations 22, 23, and 26 in equation 20, we can express the scattered field E_2 as:

$$E_2 = \frac{E_{o1} i k e^{i k R_o}}{4 \pi R_o} \int_{-X}^X \int_{-Y}^Y (a h'_x + c h'_y - b) e^{i \mathbf{v} \cdot \mathbf{r}} dx dy \quad (27)$$

where

$$\mathbf{v} = (v_x, v_y, v_z)$$

$$\begin{aligned}
&= k(\sin\theta_i - \sin\theta_r \cos\phi_r)x \\
&\quad + k(\sin\theta_r \sin\phi_r)y - k(\cos\theta_i + \cos\theta_r)z \\
a &= (1-F)\sin\theta_i + (1+F)\sin\theta_r \cos\phi_r \\
b &= (1+F)\cos\theta_r - (1-F)\cos\theta_i \\
c &= -(1+F)\sin\theta_r \sin\phi_r
\end{aligned} \quad (28)$$

If the admittance of the surface is finite, we can see from equations 24 and 25 that the Fresnel reflection coefficient F is an involved function of the local angle of incidence θ_i . For a rough surface, the local orientation will depend on the local slope of the surface. In other words, the factors a , b , and c in equation 27 will not be constant over the surface area. Therefore, for finite admittance, the integral becomes very cumbersome to evaluate, and no solution to the scattering problem is known that is general and exact at the same time. This leads us to our next assumption: the surface medium is considered to be a perfect conductor, i.e. $Y \rightarrow \infty$. From equations 24 and 25, we then see that:

$$F_{para} = 1, \text{ and } F_{perp} = -1 \quad (29)$$

and the terms a , b , and c in equation 27 are independent of x and y . We also assume the incident wave to be of perpendicular polarization, i.e. $F = F_{perp} = -1$.

The terms h'_x and h'_y in equation 27 denote the slopes of the surface $h(x, y)$ in the x and y directions, respectively. If the surface is perfectly smooth, we see that $h = 0$, $h'_x = 0$, and $h'_y = 0$. A perfectly smooth surface will reflect light only in the specular direction $\theta_r = \theta_i$, and for this direction we see that $v \cdot r = 0$. Therefore, the field E_{2ss} scattered in the specular direction by a smooth perfectly conducting surface is:

$$E_{2ss} = \frac{E_{o1} i k e^{i k R_o}}{4 \pi R_o} \int_{-X}^X \int_{-Y}^Y 2 \cos\theta_i dx dy \quad (30)$$

or:

$$E_{2ss} = \frac{E_{o1} i k e^{i k R_o} \cos\theta_i A}{2 \pi R_o} \quad (31)$$

The magnitude of the field scattered in the specular direction by the smooth perfectly conducting surface is:

$$|E_{2ss}| = \frac{E_{o1} A \cos\theta_i}{\lambda R_o} \quad (32)$$

We see that for a perfectly smooth surface, the scattered field is obtained with ease. However, a perfectly smooth surface is only the limiting case of a rough one. We will assume that our surface has random irregularities. By using a statistical model for the irregularities, we can predict the reflection characteristics of the surface. The uncertainty in height of a surface point can be described by a probability

distribution function. Though Beckmann and Spizzichino have discussed a variety of distributions, they consider the normal distribution to be the most important and typical of a rough surface.

The normal height distribution model was described in the previous section. The surface height has the mean value $\langle h \rangle = 0$, standard deviation σ_h , and correlation distance T . The normal distribution $p_h(h)$ is given by equation 4 and the autocorrelation function $C(\tau)$ by equation 5. Since h and the scattered field E_2 are related by equation 27, the statistics of E_2 can be determined from the statistics of h . Beckmann and Spizzichino have derived in detail the mean field and mean power scattered by the surface in an arbitrary direction for any given angle of incidence. They normalize the field and introduce the scattering coefficient $\rho = E_2/E_{2ss}$, and present a detailed derivation of the first and second order statistics of ρ . This normalization gets rid of the factor in front of the integral in equation 27 and helps reduce the number of terms involved in the derivation. Since E_{2ss} is constant, ρ and E_2 are proportional to each other, and the statistics of E_2 can be determined from those of ρ . It turns out that the mean field $\langle E_2 \rangle$ will be non-zero in the specular direction ($\theta_r = \theta_i$) but will tend rapidly toward zero as θ_r deviates from the specular direction. Since $\langle E_2 \rangle$ is a complex quantity, a physical interpretation of its dependency on θ_i and θ_r is not obvious. For example, it does not follow from $\langle E_2 \rangle = 0$ that $\langle |E_2| \rangle = 0$. Therefore, Beckmann and Spizzichino have only used $\langle E_2 \rangle$ as a stepping stone to derive the mean scattered power $\langle E_2 E_2^* \rangle = \langle |E_2|^2 \rangle$. For an incidence angle θ_i , the mean power scattered in the direction (θ_r, ϕ_r) by a rough surface, whose height h is normally distributed with mean value $\langle h \rangle = 0$, standard deviation σ_h , and correlation distance T , is given by:

$$\begin{aligned}
\langle E_2 E_2^* \rangle &= \frac{E_{o1}^2 A^2 \cos^2\theta_i}{\lambda^2 R_o^2} e^{-g} \left(\rho_o^2 \right. \\
&\quad \left. + \frac{\pi T^2 D^2}{A} \sum_{m=1}^{\infty} \frac{g^m}{m! m} e^{-v_{xy}^2 T^2 / 4m} \right) \quad (33)
\end{aligned}$$

where

$$g = \left(2\pi \frac{\sigma_h}{\lambda} (\cos\theta_i + \cos\theta_r) \right)^2 \quad (34)$$

$$\rho_o = \text{sinc}(v_x X) \text{sinc}(v_y Y) \quad (35)$$

$$D = \left(\frac{1 + \cos\theta_i \cos\theta_r - \sin\theta_i \sin\theta_r \cos\phi_r}{\cos\theta_i (\cos\theta_i + \cos\theta_r)} \right) \quad (36)$$

$$v_{xy} = \sqrt{v_x^2 + v_y^2} \quad (37)$$

In the previous section, the Rayleigh criterion was described to illustrate how the roughness of a surface is

related to the wavelength of incident light. We see from equation 34 that the factor g in equation 33 is proportional to the square of σ_h/λ . Therefore, g represents the roughness of the surface, and the three cases $g \ll 1$, $g \approx 1$, and $g \gg 1$ correspond to *smooth surfaces*³, *moderately rough surfaces*, and *rough surfaces*, respectively. It is important to note that the model under consideration only attempts to describe the reflection mechanism that is often referred to by the vision research community as "specular reflection". As seen from equation 33, the mean scattered power is the sum of two terms. The first term, $e^{-g} \rho_o^2$, is the *specular spike* component of the specular reflection. It is seen from equation 35 that when the surface dimensions are small, ρ_o becomes a very sharp function of θ_i and θ_r and is equal to zero for all scattering directions except a very narrow range around the specular direction. Since the mean slope of the surface is constant and is independent of the roughness of the surface, a privileged scattering in the specular direction is expected. The second term in equation 33 corresponds to the *specular lobe*⁴, i.e. the diffusely scattered field that results from the roughness of the surface. As we will see shortly, the specular lobe component is distributed around the specular direction. For a perfectly smooth surface, $g = 0$ and the specular lobe vanishes, while the specular spike is strong. As the roughness measure g increases, the spike component shrinks rapidly, while the lobe component increases in magnitude. The exponential series given by the summation in the lobe component may be approximated for smooth ($g \ll 1$) and very rough ($g \gg 1$) surfaces. The approximations result in simpler expressions for the scattered power for these two extreme surface conditions:

$$\begin{aligned} \langle E_2 E_2^* \rangle_{\text{smooth}} &= \frac{E_{o1}^2 A^2 \cos^2 \theta_i}{\lambda^2 R_o^2} e^{-g} (\rho_o^2 \\ &+ \frac{\pi T^2 D^2 g}{A} e^{-v_{xy}^2 T^2 / 4}) \end{aligned}$$

when $g \ll 1$ (38)

$$\begin{aligned} \langle E_2 E_2^* \rangle_{\text{rough}} &= \frac{E_{o1}^2 A \cos^2 \theta_i \pi T^2 D^2}{\lambda^2 R_o^2 v_z^2 \sigma_h^2} \exp \left(\frac{-v_{xy}^2 T^2}{4 v_z^2 \sigma_h^2} \right) \end{aligned}$$

when $g \gg 1$ (39)

The above equations for scattered power represent the Beckmann-Spizzichino reflectance model. Before

³We define a smooth surface as one that is either perfectly smooth or "slightly" rough.

⁴Beckmann and Spizzichino have referred to this component as the "diffuse" component. The term "diffuse" has historically been used by the vision community to describe the reflection component that results from other mechanisms such as multiple reflections and internal scattering. To avoid confusion we will refer to the diffuse component of specular reflection as the specular lobe.

we study the reflectance curves predicted by this model, it is important to understand the conditions that ensure the validity of the model. We therefore summarize the assumptions we have made during the derivation of the model and discuss the restrictions imposed by these assumptions.

4.1.3 Assumptions and Related Comments

- The surface height is assumed to be normally distributed. However, Beckmann and Spizzichino have derived reflectance models for surfaces with other height distributions, and also surfaces with periodic profiles.
- The radius of curvature of surface irregularities is large compared to the wavelength of incident light (Kirchoff's assumption). This assumption is required to approximate the electromagnetic field and its normal derivative on the surface. The approximation will break down if the surface irregularities include sharp edges or sharp points.
- The surface is assumed to be a perfect conductor. This assumption forces the quantities a , b , and c in equation 27 to be constants, thus making it easier to evaluate the Helmholtz integral. Beckmann and Spizzichino claim that this assumption is not as severe as it may first appear and that surface roughness has a greater effect on the scattered field than the electrical properties of the surface medium. Moreover, it is possible to approximate the scattered field and power for finite conductors by averaging the Fresnel coefficient F over the entire surface area and using the resultant value $\langle F \rangle$ as a constant in the Helmholtz integral. This way the mean field and mean power scattered by a finite conductor are found [1] to be

$$\langle E_2 \rangle_f = \langle F \rangle \langle E_2 \rangle_\infty \quad (40)$$

$$\langle E_2 E_2^* \rangle_f = \langle F F^* \rangle \langle E_2 E_2^* \rangle_\infty, \quad (41)$$

where the indices f and ∞ denote finite and infinite conductivity, respectively.

- We have ignored the *masking* and *shadowing* of surface points by adjacent surface points. Adjacent points may obstruct either the wave incident at a given point or the waves scattered from it. Clearly, these effects are functions of the angles of incidence and reflection. It is possible to compensate for the shadowing and masking effects by replacing the height function $h(x, y)$ by $S(x, y)h(x, y)$, where $S(x, y)$ is the shadowing function [28] that tends toward unity for surface points that are illuminated and zero for those that are not.

- We have assumed that the incident wave is reflected only once and does not bounce between surface points before it is scattered in the direction of the observation point P . Without this assumption it would be very difficult to compute the scattered field; no closed-form solution that takes *multiple scatterings* into account is known at the present time.
- The incident wave is assumed to be perpendicularly polarized. The mean field and power can also be determined for parallel polarization. Beckmann and Spizzichino have also discussed possible approaches to solving the scattering problem when the polarization vector \mathbf{e}_1 of the incident wave is neither parallel nor perpendicular to the plane of incidence.
- The incident wave is assumed to be a plane wave. This assumption is reasonable when the source is at a great distance from the surface, relative to the physical dimensions of the surface. If the source is relatively close to the surface, the incident waves must be considered to be spherical waves. We have also assumed the observation point to be sufficiently far removed from the surface to regard the scattered waves as plane waves.

4.1.4 Surface Radiance and Image Irradiance from Scattered Field

The physical optics reflection model predicts the mean field and mean power scattered by a rough surface. We are interested in the radiance of the surface since we know that radiance can be related to image irradiance [10]. Radiance was defined in Section 2 as:

$$L_r = \frac{d^2 \Phi_r}{d\omega_r dA_s \cos \theta_r} \quad (42)$$

Consider the image formation geometry shown in Figure 10. For convenience, we will use the areas and solid angles shown in the figure to determine the surface radiance. The surface element dA_s is projected by the lens onto an area dA_{im} on the image plane. Since the solid angles subtended from the center P of the lens by both areas dA_s and dA_{im} are equal, we can relate the two areas as:

$$dA_s = \frac{dA_{im} \cos \gamma}{\cos \theta_r} \left(\frac{z}{f} \right)^2 \quad (43)$$

As the viewing direction θ_r changes, we see that the surface area dA_s that is projected onto the same image element (pixel) area changes as a function of θ_r . Since the image element area dA_{im} is constant for a given sensor, the surface area dA_s must be determined from dA_{im} . All light rays radiated from dA_s that are incident on the lens area dA_l are

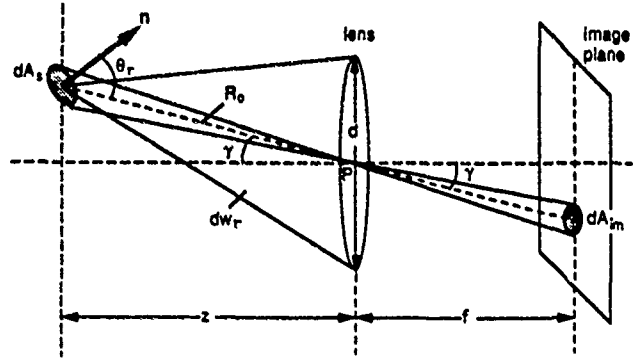


Figure 10: Image formation: light waves radiated by the surface area dA_s and gathered by the lens are projected onto an area dA_{im} on the image plane. Adapted from [10].

projected onto the image area dA_{im} . Therefore, $d\omega_r$ in equation 42 corresponds to the solid angle subtended by the lens when viewed from the area dA_s , and is determined as:

$$d\omega_r = \frac{dA_l \cos \gamma}{R_o^2} \quad (44)$$

The flux $d^2 \Phi_r$ in equation 42 is the energy of light received by the lens area dA_l , and can be determined from equation 15 as:

$$d^2 \Phi_r = S_a dA_l \cos \gamma = \frac{1}{2} \sqrt{\frac{\mu}{\epsilon}} < E_2 E_2^* > dA_l \cos \gamma \quad (45)$$

By substituting equations 43, 44, and 45 into equation 42, we obtain:

$$L_r = \frac{1}{2} \sqrt{\frac{\mu}{\epsilon}} \frac{R_o^2 f^2}{z^2 dA_{im} \cos \gamma} < E_2 E_2^* > \quad (46)$$

It is not possible to determine the exact value of the radiance from the statistics of the scattered field. The radiance L_r in the above equation is actually the mean (expected) radiance, $< L_r >$. The mean scattered power $< E_2 E_2^* >$ was determined as an integral over the entire area of the surface. In Figure 10, we see that the image element dA_{im} receives light radiated only by the surface element dA_s and, therefore, the mean scattered power must be computed as an integral over the surface area $A = dA_s$. Since the image element area dA_{im} is constant for all viewing directions θ_r , the area of integration dA_s is determined by equation 43. Thus, for a given incidence angle θ_i , the radiance in the direction (θ_r, ϕ_r) of a rough surface, whose height h is normally distributed with mean value $< h > = 0$, standard deviation σ_h , and correlation distance T , is given as:

$$L_r = \sqrt{\frac{\mu}{\epsilon}} \frac{E_{oi}^2 \cos^2 \theta_i}{2 \lambda^2} e^{-g} \left(\left(\frac{z}{f} \right)^2 \frac{dA_{im} \cos \gamma}{\cos^2 \theta_r} \rho_o^2 \right)$$

$$+ \frac{\pi T^2 D^2}{\cos \theta_r} \sum_{m=1}^{\infty} \frac{g^m}{m!m} e^{-v_{xy}^2 T^2 / 4m} \quad (47)$$

Similarly, from equations (38) and (39), the surface radiance for smooth and rough surfaces may be written as

$$L_{r\text{smooth}} = \sqrt{\frac{\mu}{\epsilon}} \frac{E_{oi}^2 \cos^2 \theta_i}{2 \lambda^2} e^{-g} \left(\left(\frac{z}{f} \right)^2 \frac{dA_{im} \cos \gamma}{\cos^2 \theta_r} \rho_o^2 + \frac{\pi T^2 D^2 g}{\cos \theta_r} \right) \quad \text{when } g \ll 1 \quad (48)$$

$$L_{r\text{rough}} = \sqrt{\frac{\mu}{\epsilon}} \frac{E_{oi}^2 \cos^2 \theta_i}{2 \lambda^2 \cos \theta_r v_z^2 \sigma_h^2} \pi T^2 D^2 \exp \left(\frac{-v_{xy}^2 T^2}{4 v_z^2 \sigma_h^2} \right) \quad \text{when } g \gg 1 \quad (49)$$

As stated in Section 2, we can also obtain the BRDF, $f_r(\theta_i; \theta_r, \phi_r)$ of the surface from its radiance and irradiance. From Section 2, we see that surface irradiance I_s is defined as the light energy incident per unit area of the surface. If E_i is the scalar value of the incident plane wave E_1 , the surface irradiance can be obtained by once again using equation 15:

$$I_s = S_a \cos \theta_i = \frac{1}{2} \sqrt{\frac{\mu}{\epsilon}} \langle E_i E_i^* \rangle \cos \theta_i \quad (50)$$

where the term $\cos \theta_i$ accounts for the fact that the same amount of incident energy is received by a greater surface area when the angle of incidence θ_i is increased. Hence, the BRDF of the surface is determined using equations 47 and 50 as $f_r = L_r / I_s$.

Using the imaging geometry shown in Figure 10, Horn [10] has established a relationship between surface radiance L_r and image irradiance I_{im} . The image irradiance is found to be proportional to surface radiance and is given by:

$$I_{im} = L_r \frac{\pi}{4} \left(\frac{d}{f} \right)^2 \cos^4 \gamma. \quad (51)$$

When the image covers only a narrow angle of the scene, we see that $\gamma \approx 0$ and it is reasonable to assume that $\cos \gamma = 1$ in the above equations.

4.1.5 Radiance Diagrams

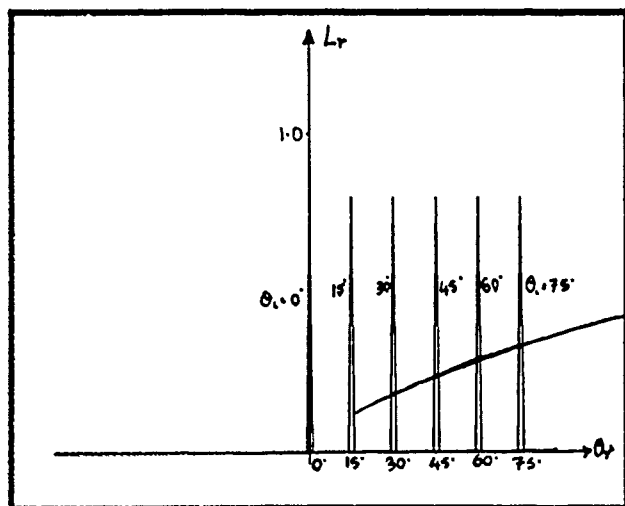
The performance of a physical optics reflectance model is usually illustrated by *scattering diagrams* [1] in which either the scattered field or the scattered power is plotted as a function of the source and viewing angles. In radiometry, surface reflectance is often represented by the BRDF $f_r(\theta_i; \theta_r, \phi_r)$ normalized by the BRDF f_{ro} in the specular direction [31]. Since we are interested in interpreting

image irradiance values, however, and since image irradiance is proportional to surface radiance, we will illustrate surface reflectance properties by *radiance diagrams*, where *absolute* surface radiance is plotted as a function of viewing angle (θ_r, ϕ_r) and incidence angle θ_i . Radiance diagrams will be plotted for different values of the surface roughness parameters. For simplicity, we will assume that the observation point P lies in the plane of incidence, i.e. $\phi_r = 0$. In this section, we will plot radiance as a function of the viewing angle θ_r for fixed values of the incidence angle θ_i . Later, we will investigate how the radiance changes as a function of θ_i , for fixed values of θ_r .

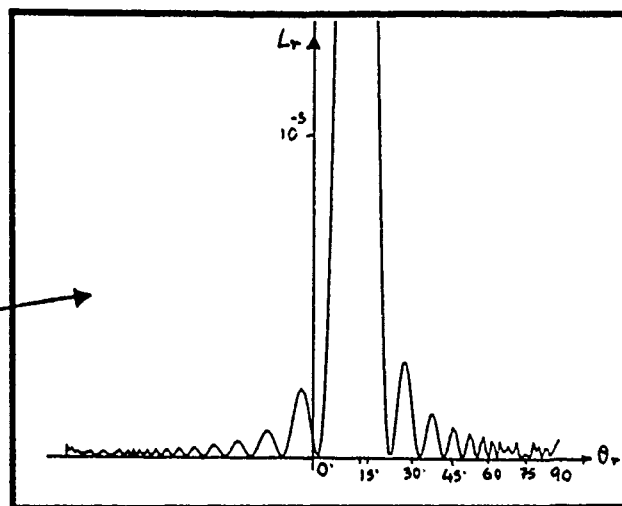
As mentioned earlier, the parameter g in equation 47 represents the roughness of the surface. We see from equation 34 that g is a function of the ratio σ_h/λ . In Section 3 we have also seen that the shape of a normally distributed surface can be represented by the ratio σ_h/T . We would like to see how the radiance diagram changes with the two ratios σ_h/λ and σ_h/T . We will vary the values of the two ratios by keeping σ_h constant and varying λ and T . Figure 11 shows radiance diagrams for different values of σ_h/λ . All the radiance diagrams are generated by using the general radiance expression given by equation 47. The specular lobe component of the radiance was computed by summing the first 100 terms of the exponential series. In Figure 11a, we see that $\sigma_h/\lambda = 0.002$, i.e. $g \approx 0$. From equation 47 we see that when $g \approx 0$, the lobe component is near zero and the spike component is dominant. The surface behaves in a mirror-like manner and reflects light only in the specular direction $\theta_r = \theta_i$. Also note that the radiance in the specular direction is constant for different values of θ_i . This is consistent with our real-world experience; when we look at a perfect mirror from the specular angle, we see a virtual image of the source. Further, the image appears the same irrespective of the angle of incidence. We have found that this mirror-like behavior is observed when $\sigma_h/\lambda < 0.025$. In Figure 11a, the spike component look like a delta function. However, from equation 35 we see that the spike component is really a *sine* function. This is seen in Figure 11b, where one of the radiance curves in Figure 11a is magnified.

As σ_h/λ is increased above the value 0.025 (Figures 11c and 11d), we find that the spike component decreases rapidly in magnitude⁵. However, the spike component is still very strong for large values of θ_r and θ_i . This is because g (equation 34) is a function not only of σ_h/λ , but also of $(\cos \theta_i + \cos \theta_r)$. Therefore, for large values of θ_i and θ_r , g approaches zero, the spike component increases, and the surface tends to behave like a mirror. However, we see that when σ_h/λ is increased further (Figures 11e and 11f),

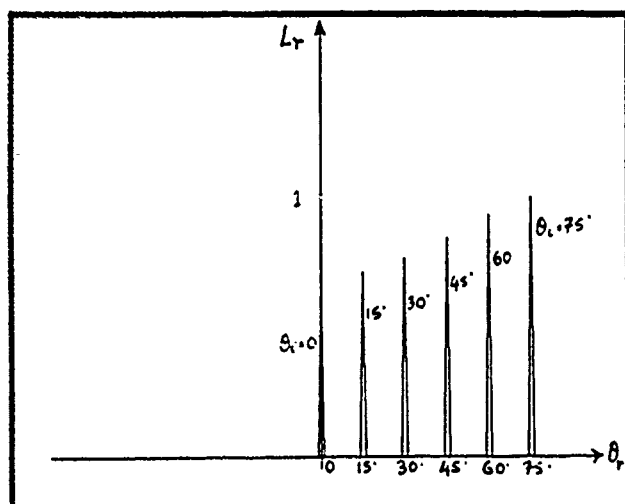
⁵If the radiance or the BRDF is normalized by the corresponding value in the specular direction, the decrease in the spike component is not observed. It is for this reason that we have chosen to plot the absolute radiance value.



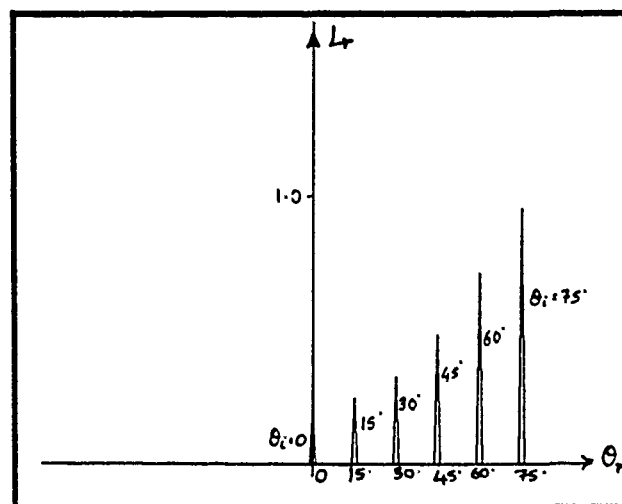
(a) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.5$



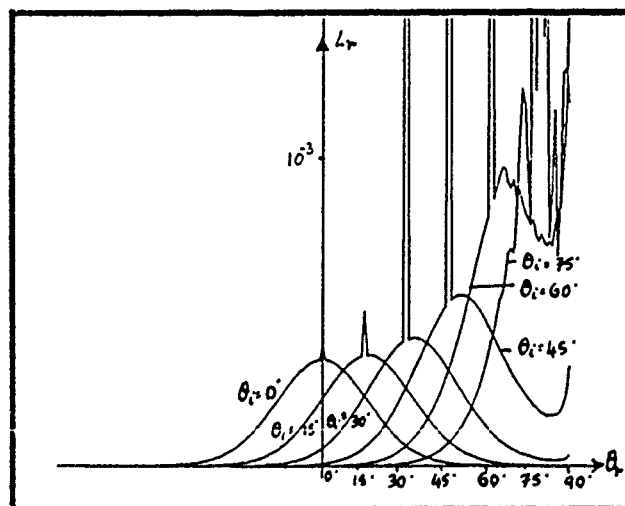
(b)



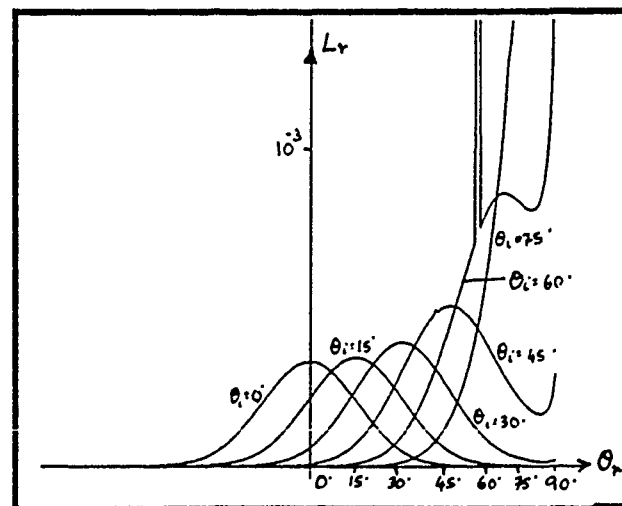
(c) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.02$



(d) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.01$



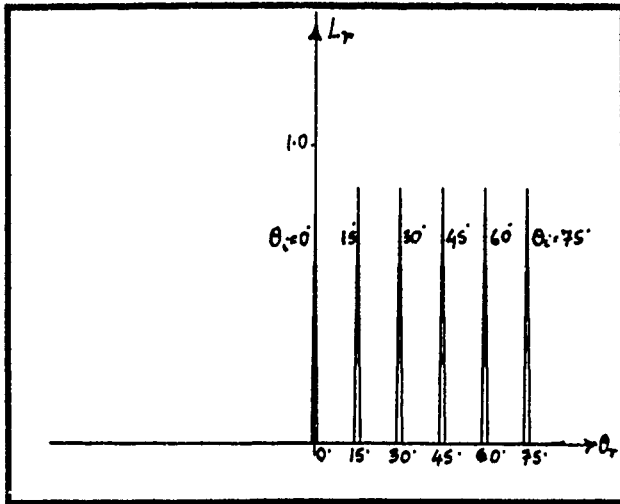
(e) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.003$



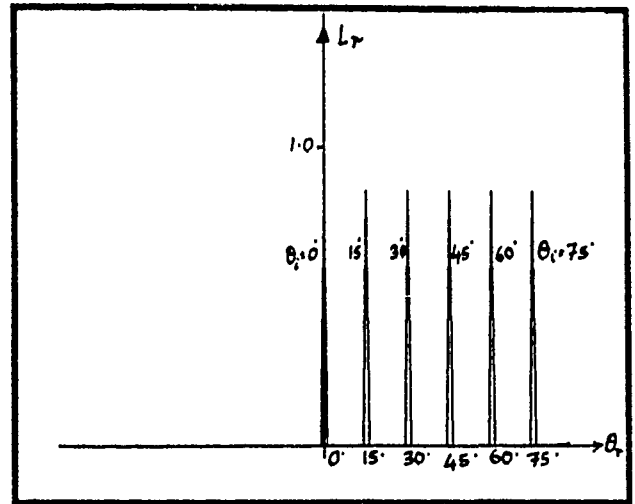
(f) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.002$

Note change of scale between (d) and (e).

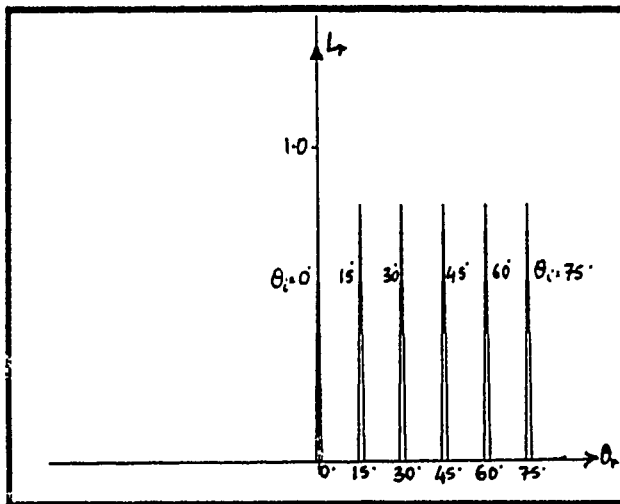
Figure 11: Radiance diagrams predicted by the Beckmann-Spizzichino model for different values of σ_h/λ .



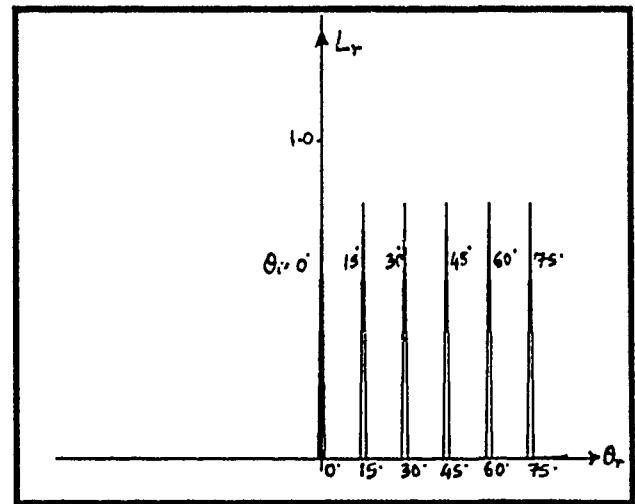
(a) $\sigma_h = 0.001$, $T = 0.05$, $\lambda = 0.5$



(b) $\sigma_h = 0.001$, $T = 0.02$, $\lambda = 0.5$

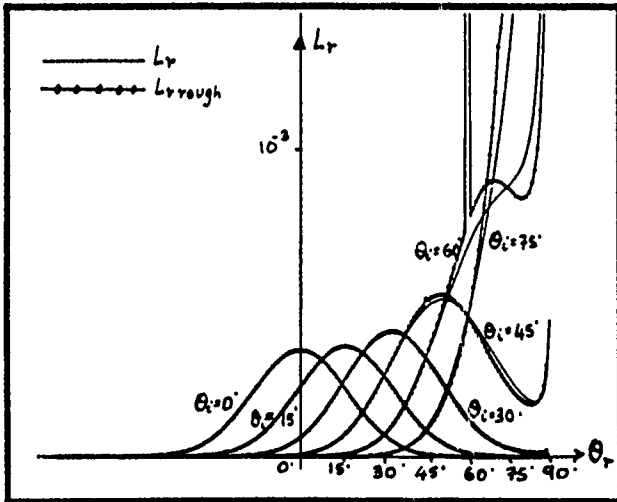


(c) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.5$

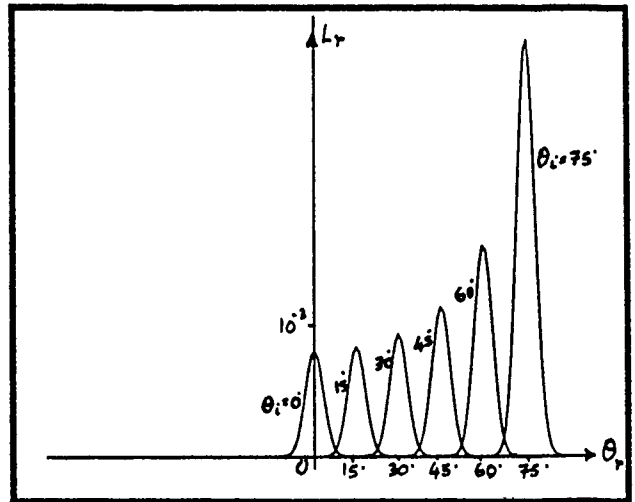


(d) $\sigma_h = 0.001$, $T = 0.006$, $\lambda = 0.5$

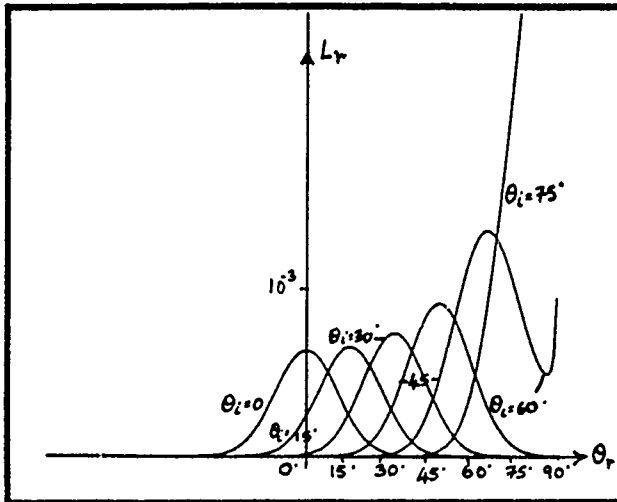
Figure 12: Radiance diagrams of the specular spike component predicted by the Beckmann-Spizzichino model for different values of σ_h/T .



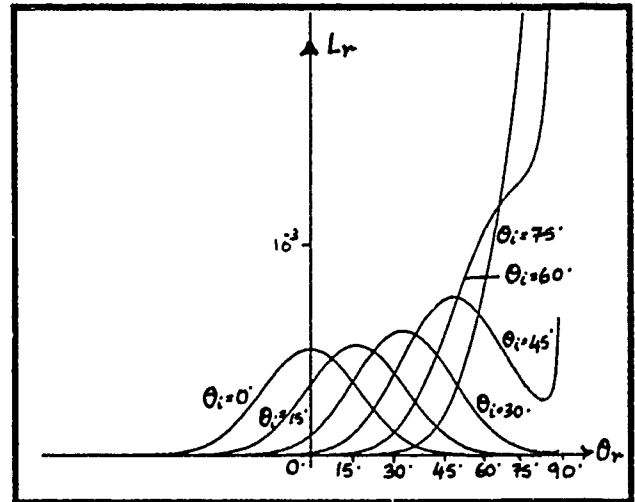
(a) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.0001$



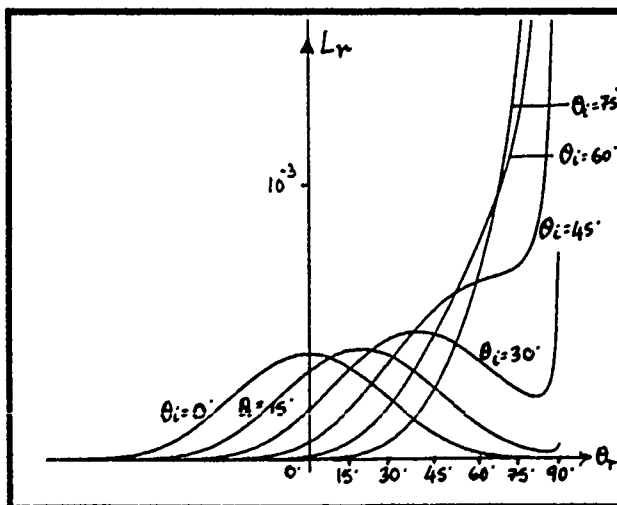
(b) $\sigma_h = 0.001$, $T = 0.05$, $\lambda = 0.0001$



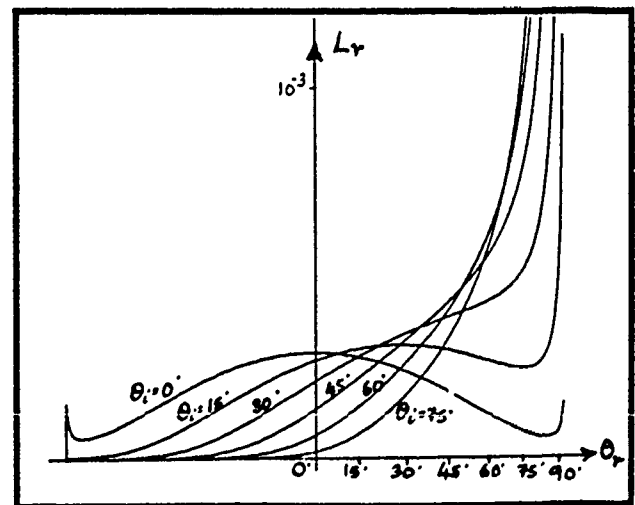
(c) $\sigma_h = 0.001$, $T = 0.015$, $\lambda = 0.0001$



(d) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.0001$



(e) $\sigma_h = 0.001$, $T = 0.007$, $\lambda = 0.0001$



(f) $\sigma_h = 0.001$, $T = 0.005$, $\lambda = 0.0001$

Figure 13: Radiance diagrams of the specular lobe component predicted by the Beckmann-Spizzichino model for different values of σ_h/T .

the spike component fades away, and the lobe component begins to dominate the radiance value. We have found that when $\sigma_h/\lambda > 1.5$, the spike component disappears, and the radiance value is determined solely by the lobe component.

Figure 12 and Figure 13 illustrate how the radiance diagram is affected by the surface roughness ratio σ_h/T . For the radiance diagrams in Figure 12, $\sigma_h/\lambda = 0.002$. We see that the spike component is unaffected by changes in the correlation distance T . In other words, for a given wavelength of incident light, the spike component would be the same for two surfaces with different shapes but the same root-mean-square height σ_h . However, in Figure 13 we see that the shape and magnitude of the lobe component are greatly dependent on the ratio σ_h/T .

In Figure 13a, we compare the radiance diagrams generated by using the general radiance expression (equation 47) and the approximate radiance expression for rough surfaces (equation 49) for $\sigma_h/\lambda = 10.0$ and $\sigma_h/T = 0.1$. We see that the expression L_{rough} approximates the lobe component of the L_r quite well, and may be used when the spike component is negligible. In Figure 13b, we see that the lobe component is sharp and concentrated around the specular direction. We have found that when $\sigma_h/T < 0.02$, the shape of the lobe component resembles that of the spike component. However, the magnitude of the lobe peak increases with the incidence angle θ_i . This effect results from the term $1/\cos\theta_i$ in equation 49. From Figure 13c-13f, we see that as the ratio σ_h/T increases, the lobe gets wider and the lobe peak decreases in magnitude. In fact, for $\sigma_h/T < 0.05$ the lobes may be approximated by Gaussian functions with mean values corresponding to the specular direction $\theta_s = \theta_r$. For larger values of σ_h/T , however, the lobes tend to peak at viewing angles greater than the specular angle; these are called *off-specular peaks*. Also note that as θ_r approaches 90 degrees, the radiance values approach infinity. By using a shadowing function, this effect can be minimized, while preserving the shape of the radiance curves for smaller values of θ_r .

4.2 Geometrical Optics Model

An outstanding feature of visible light is its short wavelength. Often, the wavelength of incident light is far shorter than the physical dimensions of the surface imperfections it encounters, and in such cases it is possible to solve the problem of reflection in an approximate way. The approximation that is valid for short wavelengths of light is known as *geometrical optics*, and it allows us to treat the reflection problem in a way far simpler than the physical optics approach of solving Maxwell's equations.

In this section, we will discuss the Torrance-Sparrow model, which uses geometrical optics to describe

the specular reflection mechanism. To their specular reflection model, Torrance and Sparrow have appended the Lambertian model to account for internal scattering and multiple reflection mechanisms. We will very briefly describe the Lambertian model⁶ and proceed to explain the Torrance-Sparrow model, once again highlighting the important steps and assumptions. Later, we will present radiance diagrams predicted by the Torrance-Sparrow model for different surface roughness values, and compare it to the Beckmann-Spizzichino physical optics model.

4.2.1 Lambertian Model

Lambert [15] was the first to investigate the mechanisms underlying diffuse reflection. Surfaces that satisfy Lambert's law appear equally bright from all directions. In other words, the radiance of a Lambertian surface is independent of the viewing direction. Broadly speaking, there are two mechanisms that produce Lambertian reflection. In one case, the light rays that impinge on the surface are reflected many times by surface undulations before they are scattered into space, as shown in Figure 14a. If these *multiple reflections* occur in a random manner, the incident energy is distributed in all directions, resulting in diffuse reflection. Another mechanism leading to Lambertian reflection is the *internal scattering* of light rays. In this case, the light rays penetrate the surface and encounter microscopic inhomogeneities in the surface medium, as shown in Figure 14b. The light rays are repeatedly reflected and refracted at boundaries between regions of differing refractive indices. Some of the scattered rays find their way to the surface with a variety of directions, resulting in diffuse reflection. When diffuse reflection produced by either or both of the above mechanisms produce constant surface radiance in all directions, we have Lambertian reflection.

The surface radiance L_r of a Lambertian surface is proportional to the irradiance I_s (incident energy per unit area) of the surface. Consider an infinitesimal surface area dA_s illuminated by an infinitesimal source area dA_i , as shown in Figure 15. The flux incident on dA_s may be determined from the source radiance L_i as:

$$d^2\Phi_i = L_i d\omega_s dA_i. \quad (52)$$

From the solid angles subtended by the surface and source areas, we obtain:

$$dA_i = d\omega_i r^2, \quad (53)$$

$$d\omega_s = \frac{dA_s \cos\theta_i}{r^2}. \quad (54)$$

⁶Lambertian reflection is normally categorized as "body" reflection rather than surface reflection. The model is discussed here only because it is used later to represent one of the primary reflection components.

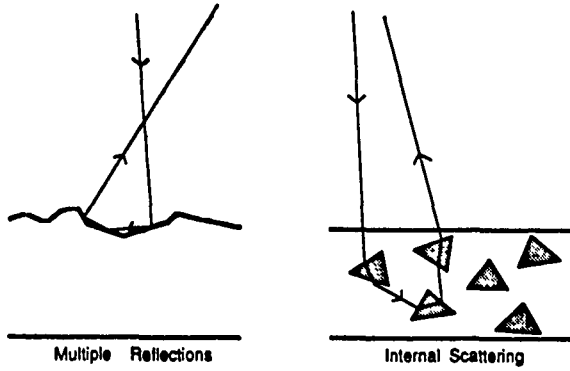


Figure 14: Diffuse reflection resulting from multiple reflection and internal scattering mechanisms.

Substituting equations 53 and 54 into equation 52, we obtain:

$$d^2 \Phi_i = L_i d\omega_i dA_s \cos \theta_i. \quad (55)$$

The surface irradiance is determined from the above equation as:

$$I_s = \frac{d^2 \Phi_i}{dA_s}, \quad (56)$$

Since surface radiance is proportional to surface irradiance, and since it is meaningful only when it attains positive values, it can be expressed as:

$$L_r = \kappa_{diff} \max[0, (L_i d\omega_i \cos \theta_i)], \quad (57)$$

where κ_{diff} determines the fraction of the incident energy that is diffusely reflected by the surface.

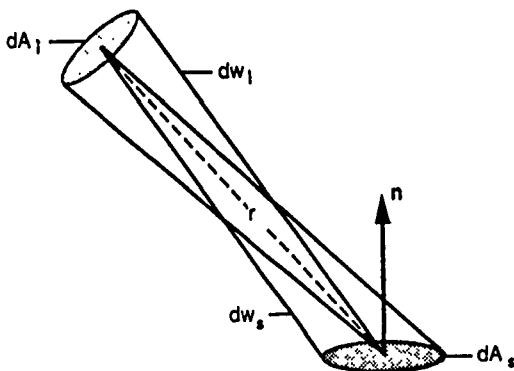


Figure 15: Dependence of the incident light energy on the source direction.

4.2.2 Torrance-Sparrow Model

The Torrance-Sparrow model was developed with the aim of describing the mechanism for specular reflection

by rough surfaces. Based on geometrical optics, this model is valid only when the wavelength of light is much smaller than the root-mean-square surface roughness. The surface is modeled as a collection of planar micro-like facets. As explained in Section 3.2, the surface has a mean surface orientation \mathbf{n} , and the slope α of each planar facet with respect to the mean orientation is described by a probability distribution. Each facet reflects incident light in the specular direction determined by its slope. Since the facet slopes are randomly distributed, light rays are scattered in various directions. Therefore, it is possible to assign a specific distribution function to the facet slopes and determine the radiance of the surface in any given direction.

Torrance and Sparrow have assumed the facet slopes to be normally distributed. Further, they have assumed the distribution to be rotationally symmetric about the mean surface normal \mathbf{n} . Hence, facet slopes may be represented by a one-dimensional normal distribution:

$$p_\alpha(\alpha) = c e^{-\frac{\alpha^2}{2\sigma_\alpha^2}}, \quad (58)$$

where c is a constant, and the facet slope α has mean value $\langle \alpha \rangle = 0$ and standard deviation σ_α . As we have stated earlier, for this surface model, roughness is represented by the parameter σ_α .

Consider the geometry shown in Figure 16. The surface area dA_s is located at the origin of the coordinate frame, and its surface normal points in the direction of the z -axis. The surface is illuminated by a beam of light that lies in the x - z plane and is incident on the surface at an angle θ_i . We are interested in determining the radiance of the surface in the direction (θ_r, ϕ_r) . Only those planar micro-facets whose normal vectors lie within the solid angle $d\omega'$ are capable of specularly reflecting light flux that is incident at the angle θ_i into the infinitesimal solid angle $d\omega_r$. From the angles θ_i , θ_r , and ϕ_r , we can determine the local angle of incidence θ_i' and slope α of the reflecting facets:

$$\theta_i' = \frac{1}{2} \cos^{-1} (\cos \theta_r \cos \theta_i - \sin \theta_r \sin \theta_i \cos \phi_r), \quad (59)$$

$$\alpha = \cos^{-1} (\cos \theta_i \cos \theta_i' + \sin \theta_i \sin \theta_i' \cos(\sin^{-1}(\sin \phi_r \sin \theta_r / \sin 2\theta_i'))). \quad (60)$$

The number of facets per unit area of the surface that are oriented within the solid angle $d\omega'$ is equal to $(p_\alpha(\alpha) d\omega')$. Therefore, the number of facets in the surface area dA_s that are oriented within $d\omega'$ is equal to $(p_\alpha(\alpha) d\omega' dA_s)$. Let a_f be the area of each facet. Then, the area of points in dA_s that will reflect light from the direction θ_i into the solid angle $d\omega_r$ is equal to $(a_f p_\alpha(\alpha) d\omega' dA_s)$. All the reflecting facets

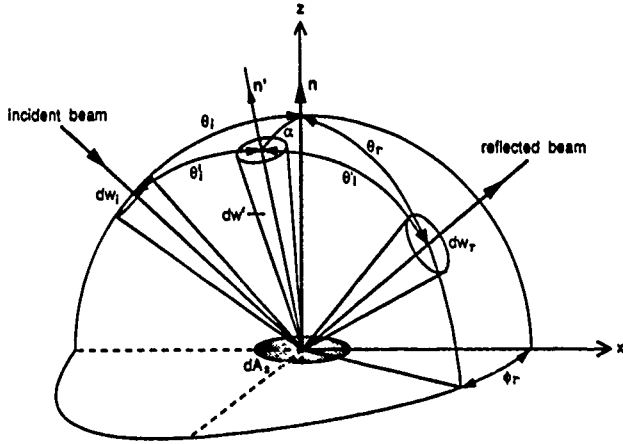


Figure 16: Coordinate system used to derive the Torrance-Sparrow model.

are assumed to have the same local angle of incidence, θ'_i . From equation 55, the flux incident on the set of reflecting facets is determined as:

$$d^2\Phi_i = L_i d\omega_i (a_f p_\alpha(\alpha) d\omega' dA_s) \cos\theta'_i. \quad (61)$$

The fraction of incident light that is reflected by each planar facet is determined by the Fresnel reflection coefficient. The Fresnel coefficients $F_{para}(\theta'_i, \eta')$ and $F_{perp}(\theta'_i, \eta')$ determine the electromagnetic field reflected in the specular direction by a planar surface when the incident wave is of parallel and perpendicular polarization, respectively. In this section, however, we are interested in the reflected flux, i.e. the energy flowing through a unit area. The reflection coefficients for energy reflectance may be determined from those for field reflectance as:

$$\begin{aligned} F'_{para}(\theta'_i, \eta') &= |F_{para}(\theta'_i, \eta')|^2 \text{ and,} \\ F'_{perp}(\theta'_i, \eta') &= |F_{perp}(\theta'_i, \eta')|^2. \end{aligned} \quad (62)$$

Let us assume that the polarization vector e_1 of the incident light wave lies outside the plane of incidence, and let h and v represent the magnitudes of the resolved components of e_1 in the parallel and perpendicular polarization planes, respectively. The Fresnel coefficient $F'(\theta'_i, \eta')$ for the incident wave may be expressed as a linear combination of the Fresnel coefficients for parallel and perpendicular incident waves [26]:

$$F'(\theta'_i, \eta') = h F'_{para}(\theta'_i, \eta') + v F'_{perp}(\theta'_i, \eta'), \quad (63)$$

where

$$h, v \geq 0 \text{ and } h + v = 1. \quad (64)$$

Torrance and Sparrow have also considered the masking and shadowing of one micro-facet by adjacent facets. Adjacent facets may obstruct flux incident upon a given facet or the flux reflected by it. In order to compensate for these effects, the *geometrical attenuation factor*⁷ $G(\theta_i, \theta_r, \phi_r)$ is introduced. The obstruction of incident or reflected light will depend on the angle of incidence and the angles of reflection. Each facet is assumed to be one side of a V-groove cavity, and light rays are assumed to be reflected only once. A detailed derivation of $G(\theta_i, \theta_r, \phi_r)$ is given in [31], and the final expression is found to be:

$$G(\theta_i, \theta_r, \phi_r) = \min \left(1, \frac{2 \cos\alpha \cos\theta_r}{\cos\theta'_i}, \frac{2 \cos\alpha \cos\theta_i}{\cos\theta'_i} \right). \quad (65)$$

Taking the Fresnel reflection coefficient and the geometrical attenuation factor into consideration, the flux $d^2\Phi_r$ reflected into the solid angle $d\omega_r$ may be determined from the flux $d^2\Phi_i$ incident on the reflecting facets as:

$$d^2\Phi_r = F'(\theta'_i, \eta') G(\theta_i, \theta_r, \phi_r) d^2\Phi_i. \quad (66)$$

The radiance L_r of the surface dA_s in the direction (θ_r, ϕ_r) is defined as:

$$L_r = \frac{d^2\Phi_r}{d\omega_r dA_s \cos\theta_r}. \quad (67)$$

Using equations 61 and 66, equation 67 may be written as:

$$L_r = \frac{F'(\theta'_i, \eta') G(\theta_i, \theta_r, \phi_r) L_i d\omega_i (a_f p_\alpha(\alpha) d\omega' dA_s) \cos\theta'_i}{d\omega_r dA_s \cos\theta_r} \quad (68)$$

Earlier we stated that only facets with normals that lie within the solid angle $d\omega'$ are capable of reflecting light into the solid angle $d\omega_r$. Therefore, $d\omega'$ and $d\omega_r$ are related to one another. Though Torrance and Sparrow have only used this relationship and have not derived it, we feel that it is a very important one and deserves at least an informal proof. To this end, let us consider the plane shown in Figure 17, which includes the incident and reflected beams. We will assume all incident rays of light are parallel. This assumption is valid when the source is at a large distance from the surface. We see that the areas dA_r and dA''' subtend the same solid angle from the point I , and that $IR = 2IP$. Therefore, we can relate the two areas as $dA''' = dA_r/4$. Similarly, we see that dA'' and dA''' subtend the same solid angle $d\omega'$ from the point O . Noting that $OP = \cos\theta'_i$, we can relate the two areas as $dA'' = dA'''/\cos^2\theta'_i$. Further, the area dA' is a projection of the area dA'' onto the surface of the unit sphere, i.e. $dA' = dA'' \cos\theta_i$. Using the above relations, we can relate dA' to dA_r : $dA' = dA_r/4 \cos\theta'_i$. Since $d\omega' = dA'$

⁷This factor plays the role of the shadowing function S mentioned in the previous section.

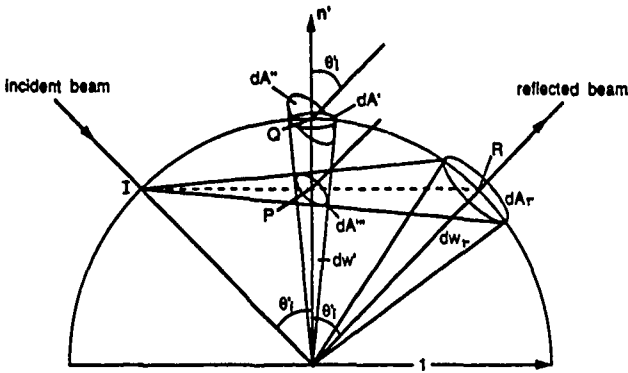


Figure 17: The source-viewer plane, illustrated to establish the relationship between dw' and dw_r .

and $dw_r = dA_r$ (areas on the unit spheres), we have:

$$dw' = \frac{dw_r}{4 \cos \theta_i'} \quad (69)$$

Hence, for a given dw_r , the shape and size of the corresponding dw' is dependent on the local angle of incidence θ_i' , which is in turn dependent on the angle of incidence θ_i and the angles of reflectance (θ_r, ϕ_r) (equation 59). Note that for a perfectly smooth surface, the parallel incident rays will be reflected in a single direction (the specular direction) and will not be scattered into a cone as shown in Figure 17. Therefore, for this limiting case, the above relationship between dw' and dw_r will not be valid.

Substituting equations 58 and 69 into equation 68, we obtain:

$$L_r = \kappa_{spec} \frac{L_i dw_i}{\cos \theta_r} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}}, \quad (70)$$

where

$$\kappa_{spec} = \frac{c a_f F'(\theta_i', \eta') G(\theta_i, \theta_r, \phi_r)}{4} \quad (71)$$

Note the similarity between the above equation and the expression for the specular lobe predicted by the Beckmann-Spizzichino model (equation 49). Thus, the Torrance-Sparrow specular reflection model describes only the lobe component of specular reflection; there is no term in the above equation that represents the spike component of specular reflection. The radiance is determined only by the roughness parameter σ_α , and unlike the Beckmann-Spizzichino model, there is no dependence on the wavelength λ of incident light. However, from the physical optics model we have seen that the spike component is significant only when $\sigma_h/\lambda < 1.5$. Torrance and Sparrow have clearly

stated that their model is only valid when $\sigma_h/\lambda \gg 1.0$. Therefore, this model must not be used to predict or interpret reflection from very smooth surfaces, i.e. when $\sigma_h/\lambda \ll 1.0$. To make their model more generic, Torrance and Sparrow have appended the Lambertian model to their specular model to account for diffuse reflection that may result from multiple reflections or internal scattering. Thus, for an angle of incidence θ_i , the radiance in the direction (θ_r, ϕ_r) of a rough surface whose facet slopes are normally distributed with standard deviation σ_α may be expressed as:

$$L_r = \kappa_{diff} \max[0, (L_i dw_i \cos \theta_i)] + \kappa_{spec} \frac{L_i dw_i}{\cos \theta_r} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} \quad (72)$$

where κ_{diff} and κ_{spec} determine the fractions of incident energy that are reflected as components of the diffuse and specular lobes, respectively. From the radiance and irradiance, the BRDF of the surface may be obtained as $f_r = L_r/I_s$. Once again, we will summarize the assumptions we have made during the derivation of this model and discuss the restrictions imposed by these assumptions.

4.2.3 Assumptions and Related Comments

- The surface is modeled as a collection of planar micro-facets, and the facet slopes are normally distributed. Other distributions, however, may be used to describe the facet slopes. For example, if the surface height is assumed to be normally distributed with standard deviation σ_h and correlation distance T , the slope distribution may be determined from the height distribution as [1]:

$$p_\alpha(\alpha) = \frac{T}{2\sigma_h \sqrt{\pi} \cos^2 \alpha} \exp\left(-\frac{\tan^2 \alpha}{2\sigma_h/T}\right) \quad (73)$$

- The size of the planar facets is much greater than the wavelength of incident light, i.e. $\sigma_h \gg \lambda$. Therefore, we can assume that the light rays are reflected by each facet in its specular direction only. Furthermore, $\sigma_h \gg \lambda$ implies that the spike component of reflection is negligible and that the model determines only the lobe component of reflection.
- The geometrical model takes the Fresnel reflection coefficient F' into account. Therefore, the polarization of incident light and the conductivity of the surface medium need not be constrained. As a result, the model is capable of predicting reflections from both conductors and dielectrics.
- Each facet comprises one side of a symmetric V-groove cavity. With this assumption, the shadowing and masking effects are compensated for by using the geometrical attenuation factor G .

- The source is assumed to be at a great distance from the surface, so that all light rays that are incident upon the surface area dA_s are nearly parallel to one another. This assumption simplifies the relationship between the solid angles $d\omega'$ and $d\omega_r$ (equation 69).
- The final model includes the Lambertian model to account for diffuse reflection mechanisms such as multiple reflection and internal scattering.

4.2.4 Radiance Diagrams

Torrance and Sparrow have evaluated the performance of their model by plotting the ratio of the BRDF in a given direction to the BRDF in the specular direction. The normalized BRDF distributions predicted by the model for a dielectric (MgO) and a conductor (Al) were found to fit the experimental data very well. We feel that plots of the normalized BRDF could lead to misinterpretation of the reflectance characteristics, however. Since image irradiance is proportional to surface radiance, we once again choose to plot absolute radiance diagrams. Since our intention is to compare the Torrance-Sparrow model with the Beckmann-Spizzichino model, we will neglect the Lambertian component of the Torrance-Sparrow model. Further, since the Torrance-Sparrow model is valid only when $\sigma_h \gg \lambda$, we will only compare it with the Beckmann-Spizzichino model for rough surfaces given by equation 49.

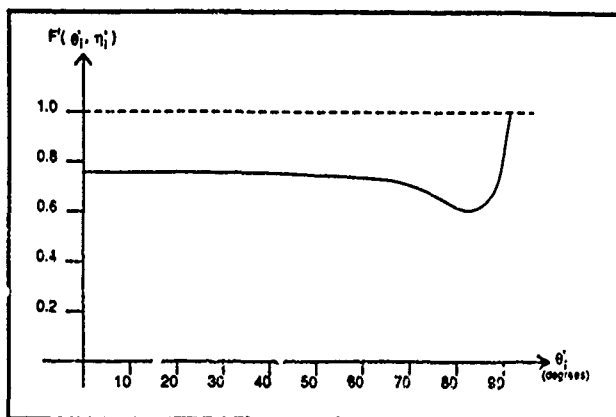


Figure 18: Typical plot of the Fresnel reflection coefficient as a function of the local incidence angle.

Consider the Fresnel coefficient $F'(\theta_i', \eta')$ and the geometrical attenuation factor $G(\theta_i, \theta_r, \phi_r)$ in equation 71. A typical plot of $F'(\theta_i', \eta')$ as a function of θ_i' is shown in Figure 18. For metals and many other surfaces, it is observed [21] that F' has a nearly constant behavior until the local angle of incidence θ_i' approaches 90 degrees. Therefore, we will assume that F' is constant with respect to θ_i and θ_r . Figure 19 shows $G(\theta_i, \theta_r, 0)$ plotted as a function of θ_r , for

different values of θ_i . We see that, for angles of incidence not near the grazing angle, G equals unity over an appreciable range of θ_r . In the following radiance diagrams, we will see that it is within these ranges of θ_r that the surface radiance attains maximum values. Therefore, we assume that $G = 1$ for all values of θ_i and θ_r . With the above two assumptions, we see that κ_{spec} is constant for all values of θ_i and θ_r .

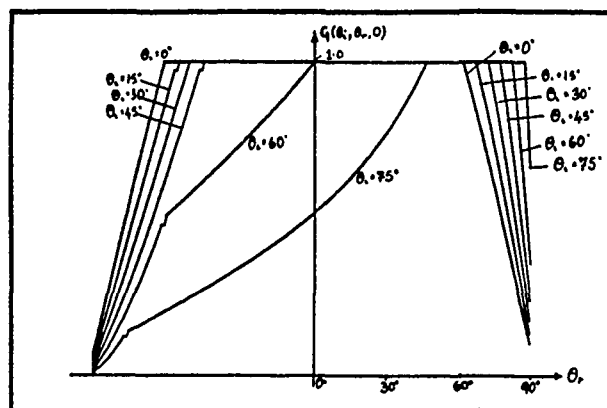
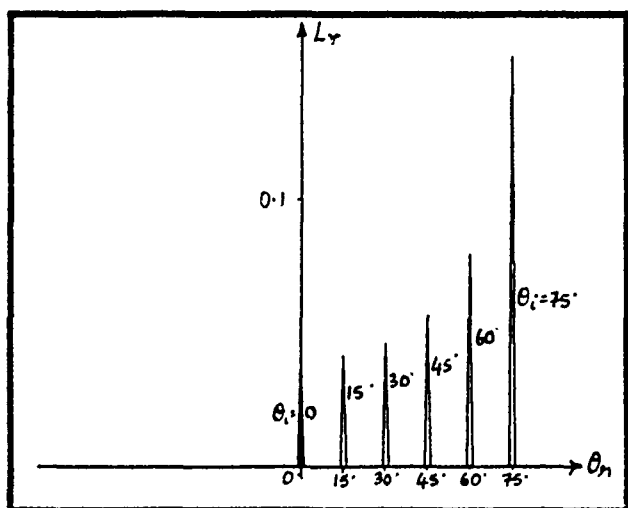


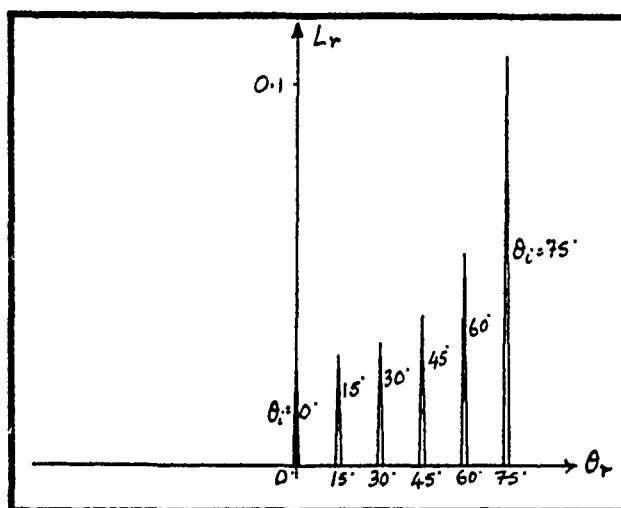
Figure 19: Geometrical attenuation function plotted as a function of the viewing angle, for different values of the incidence angle.

Figure 20 shows radiance diagrams for different values of the surface roughness parameter σ_α . Very small values of σ_α correspond to smooth surfaces, and for these values the specular lobes are similar in appearance to the specular spikes shown in Figure 11a. If the normalized BRDF is plotted rather than the absolute radiance, the lobe peaks will have constant values for all angles of incidence, and the resulting plot will appear exactly like the radiance diagram shown in Figure 11a. It is important to note that Figure 20a shows the specular lobes for a smooth surface and not the specular spikes. Therefore, the Torrance-Sparrow model is capable of predicting the specular lobe for smooth surfaces. However, for smooth surfaces, σ_h is comparable to λ , and the spike component is generally much stronger than the lobe component.

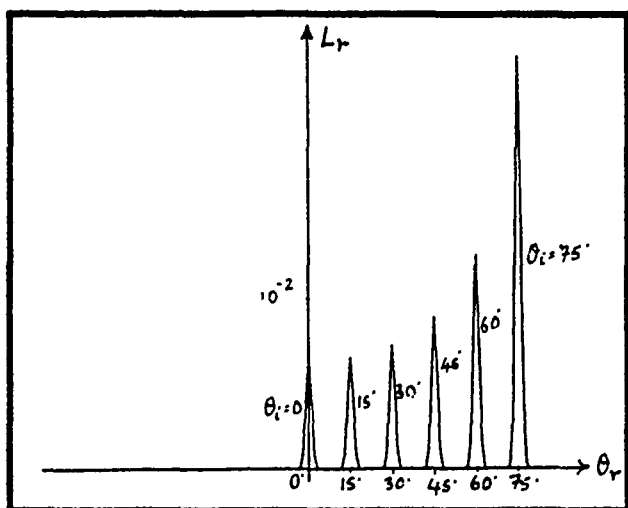
We see from Figure 20 that the peak value of the specular lobe increases in magnitude with the angle of incidence θ_i . As in the case of the physical optics model, this effect results from the term $1/\cos\theta_r$ (equation 70). It is also clear that the width of the lobe increases with the roughness parameter σ_α . In fact, for relatively small values of σ_α , the lobe may be approximate by a Gaussian function that is symmetric with respect to the specular direction. However, for higher values of σ_α (Figure 21), the lobe peak occurs at reflection angles greater than the specular angle. As with the physical optics model, these off-specular peaks result from the term $1/\cos\theta_r$ (equation 70). For large values



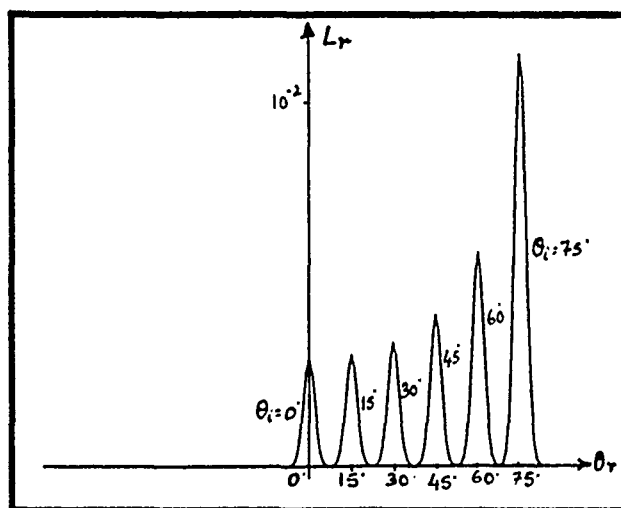
(a) $\sigma_\alpha = 0.01$



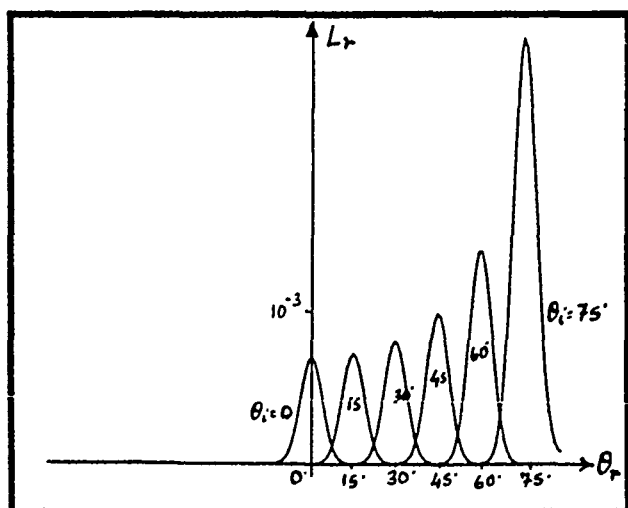
(b) $\sigma_\alpha = 0.1$



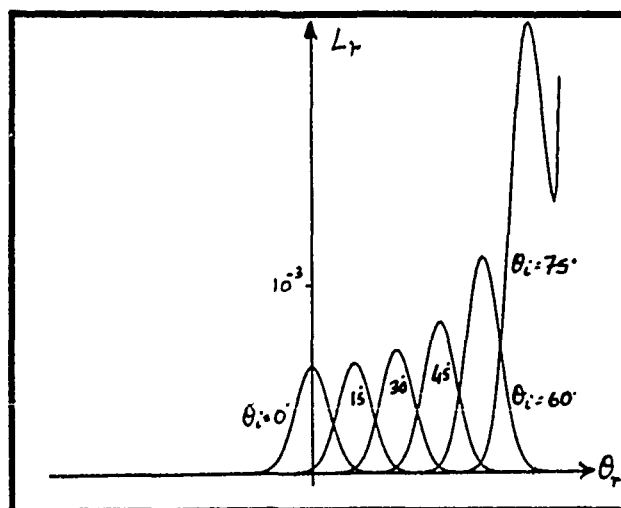
(c) $\sigma_\alpha = 0.5$



(d) $\sigma_\alpha = 1.0$

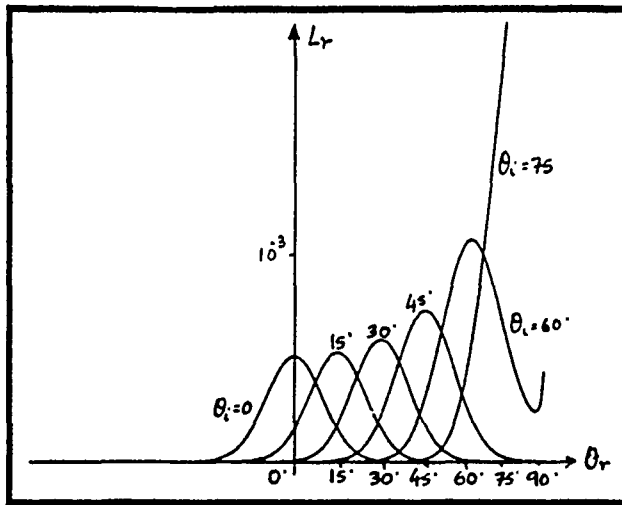


(e) $\sigma_\alpha = 2.0$

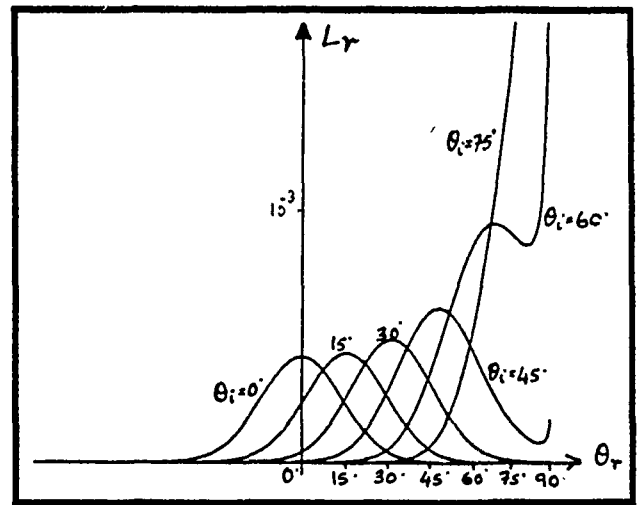


(f) $\sigma_\alpha = 3.0$

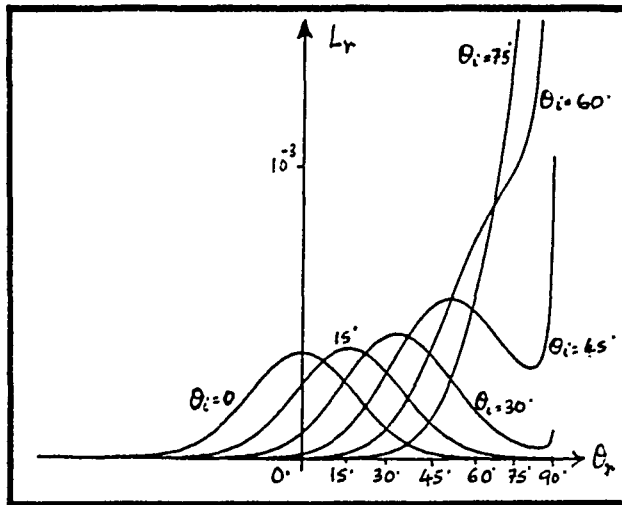
Figure 20: Radiance diagrams predicted by the Torrance-Sparrow model for different values of σ_α .



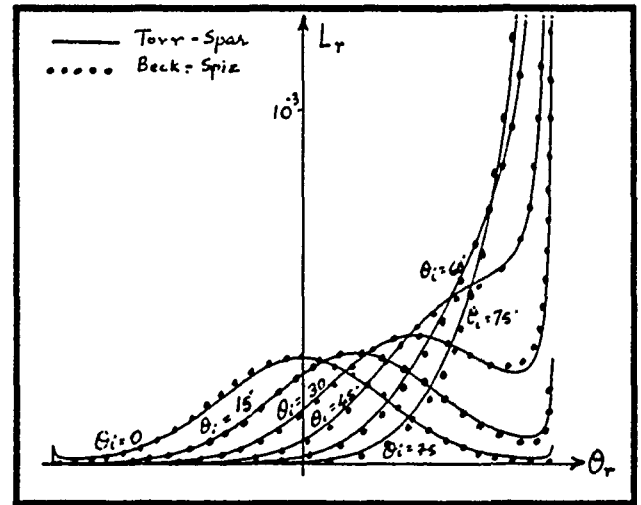
(a) $\sigma_\alpha = 5.0$



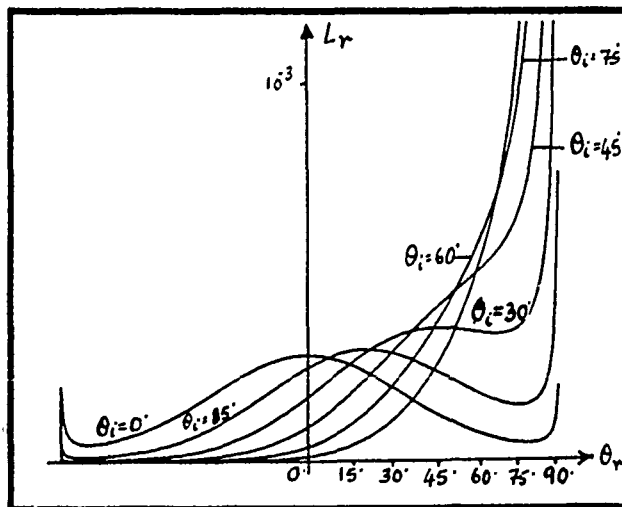
(b) $\sigma_\alpha = 7.0$



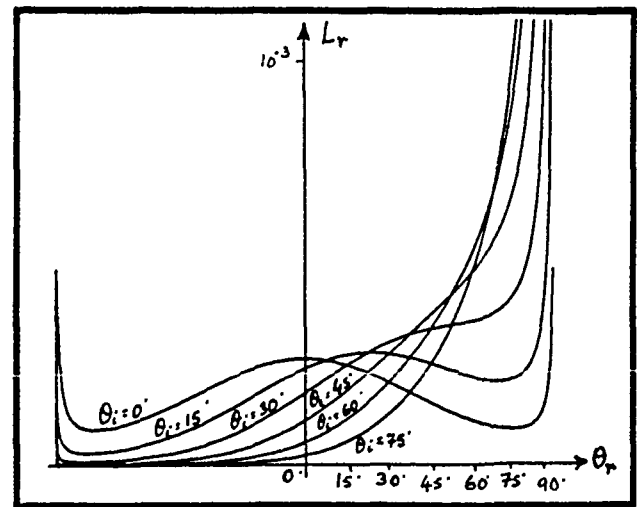
(c) $\sigma_\alpha = 9.0$



(d) $\sigma_\alpha = 13.0$



(e) $\sigma_\alpha = 15.0$



(f) $\sigma_\alpha = 17.0$

Figure 21: Radiance diagrams predicted by the Torrance-Sparrow model for different values of σ_α .

of θ_i and near-grazing values of θ_r , the radiance values approach infinity. From Figure 19 we see that G approaches zero for near-grazing values of θ_r . Torrance and Sparrow have proved that G approaches zero at a faster rate than the rate at which the plotted radiance approaches infinity. Hence, in practice, the surface radiance equals zero when $\theta_r \approx 90$ degrees. In Figure 21d, we have compared the radiance diagrams predicted by the Torrance-Sparrow model and the Beckmann-Spizzichino model. Though the two models were developed using different approaches and different surface models, we see that the resemblance between the two radiance diagrams is remarkable. In the following section, we relate the roughness parameters of the two models.

5 Observations

5.1 Primary Reflection Components

From the physical and geometrical optics reflection models, we see that surface radiance may be decomposed into three primary reflection components, namely, the diffuse lobe, specular lobe, and specular spike. Polar plots of these three components are illustrated in Figure 22. The sum of the three lobe components determines the surface radiance detected by the viewer for a fixed position of the source. The diffuse lobe is represented by the Lambertian model, and is constant with respect to the viewing direction. The specular lobe tends to be distributed around the specular direction, and has off-specular peaks for relatively large values of surface roughness. The specular spike is concentrated in a small region around the specular direction. The strengths of the specular lobe and specular spike components are related to one another. For a smooth surface, the specular spike component is many orders of magnitude greater than the specular lobe component. As the surface roughness increases, the spike component shrinks rapidly, and the specular lobe begins to dominate. We have seen from the radiance diagrams for the physical optics models that, for a given wavelength of incident light, the spike and lobe components are comparable to one another only for a small range of roughness values.

Owing to its simplicity and its conformity with experimental data [31], the specular component of the Torrance-Sparrow model may be used to approximate the specular lobe component. However, this model does not have a spike component, so the spike component of the Beckmann-Spizzichino model may be used. We see from equation 47 that the shape of the spike component is determined by the term ρ_o . Since ρ_o is a very sharp function of θ_i and θ_r , we can approximate ρ_o by a Gaussian function with low standard deviation or a double-delta function. Using the above approximations, the image irradiance equation, for fixed source direction and varying sensor direction, may

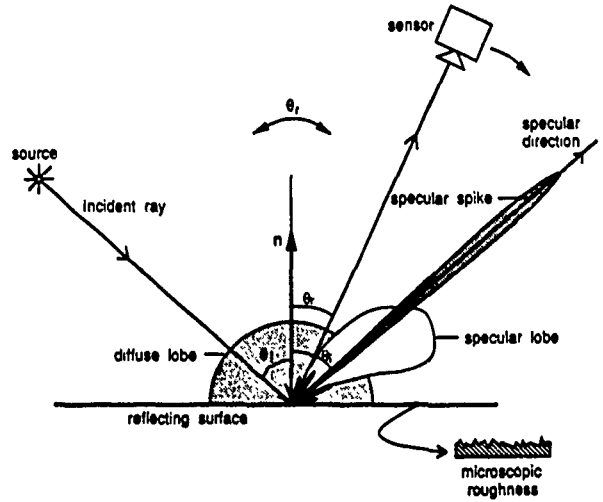


Figure 22: Polar plots of the three reflection components as functions of the viewing angle for a fixed source direction.

be written as a linear combination of the three reflection components:

$$I_{im} = C_{dl} + \frac{C_{sl}}{\cos\theta_r} \exp\left(-\frac{\alpha^2}{2\sigma_\alpha^2}\right) + C_{ss} \delta(\theta_i - \theta_r) \delta(\phi_r) \quad (74)$$

where, the constants C_{dl} , C_{sl} , and C_{ss} represent the strengths of the diffuse lobe, specular lobe, and specular spike components, respectively.

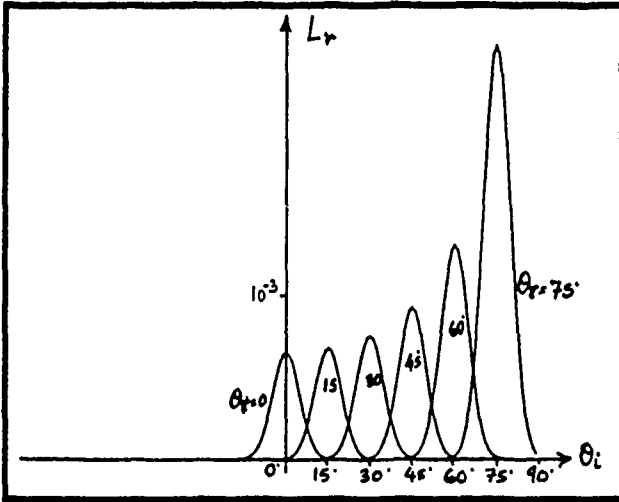
5.2 Moving Source and Fixed View

In all the radiance diagrams we have presented so far, surface radiance was plotted as a function of viewing direction θ_r , for fixed values of the incidence angle θ_i . In shape extraction techniques such as photometric stereo, structured highlight, and photometric sampling, however, images of the observed object are obtained by varying the source direction while keeping the viewing direction constant. Note that when the viewing direction is fixed, the term $1/\cos\theta_r$ in the specular component of the Torrance-Sparrow model (equation 72) is constant, and the shape of the specular lobe is dependent solely on the term

$$\exp\left(-\frac{\alpha^2}{2\sigma_\alpha^2}\right). \quad (75)$$

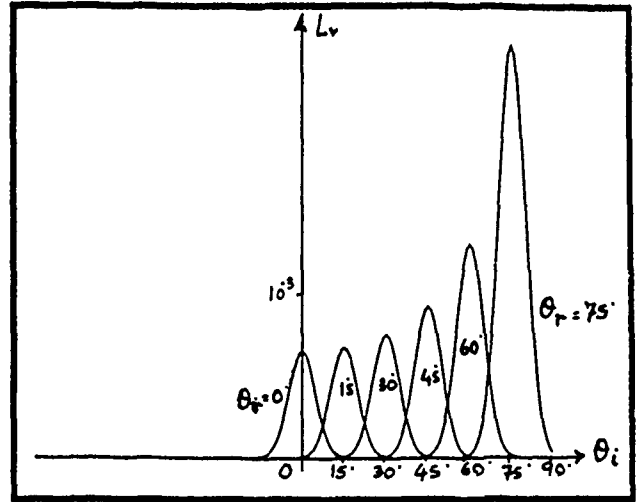
Since $\alpha = 0$ when $\theta_r = \theta_i$, the specular lobe is found to be symmetric with respect to the specular direction. A similar analysis is applicable to the physical optics model for rough surfaces (equation 49). The only term that is significantly affected by variations in θ_i is the term $e^{-v_{xy}^2 T^2 / 4v_z^2 \sigma_h^2}$.

Beckmann - Spizzichino

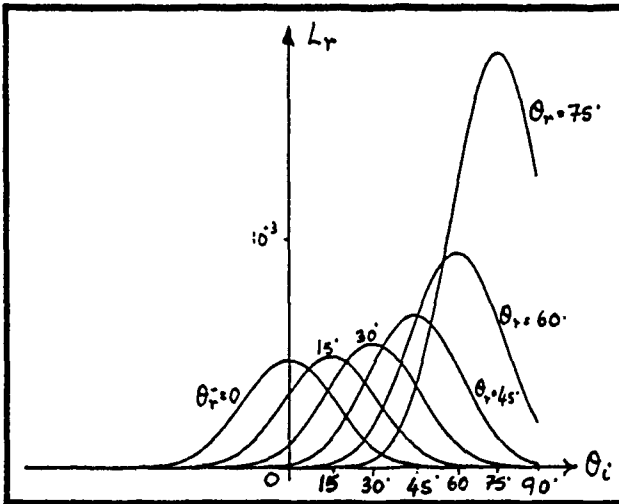


(a) $\sigma_h = 0.003$, $T = 0.01$, $\lambda = 0.001$

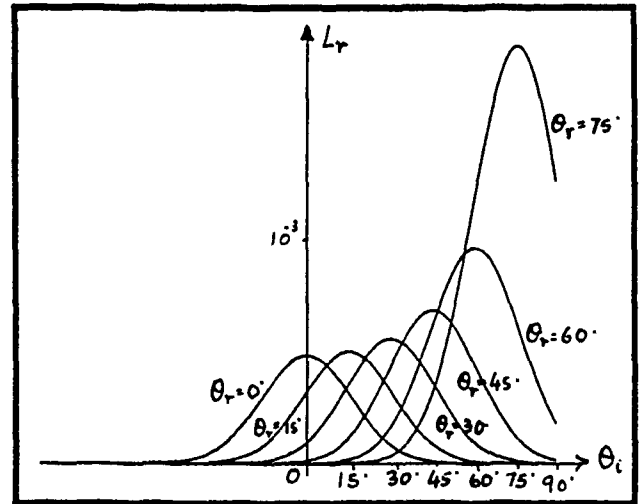
Torrance - Sparrow



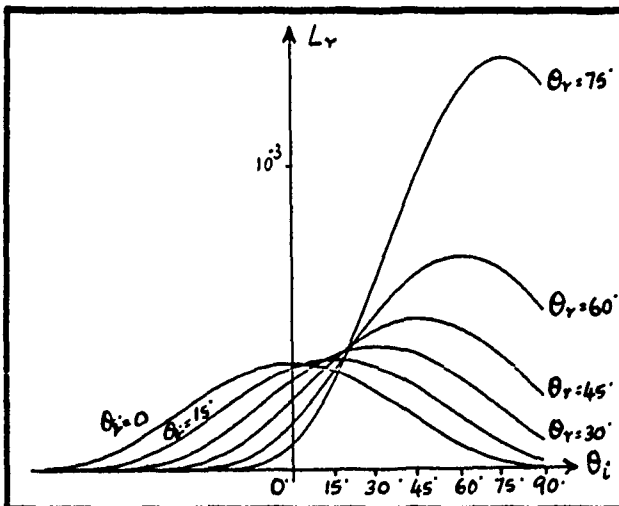
(b) $\sigma_a = 2.425$



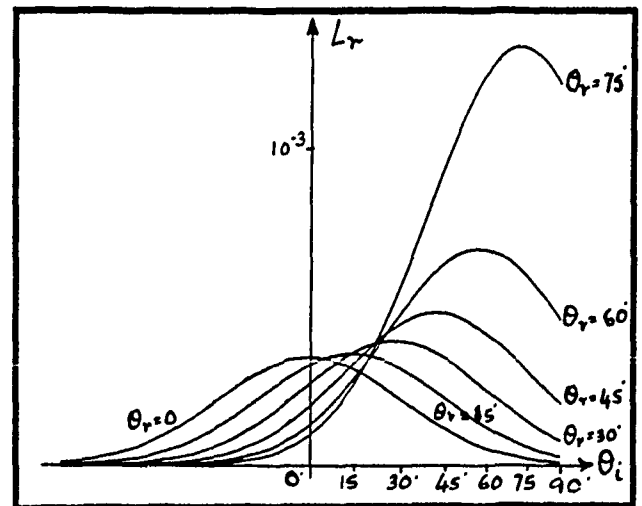
(c) $\sigma_h = 0.001$, $T = 0.01$, $\lambda = 0.001$



(d) $\sigma_a = 7.99$



(e) $\sigma_h = 0.002$, $T = 0.01$, $\lambda = 0.001$



(f) $\sigma_a = 15.416$

Figure 23: Radiance diagrams predicted by the Beckmann-Spizzichino model and the Torrance-Sparrow model. In these diagrams, radiance is plotted as a function of θ_i for fixed values of θ_r .

Further, it can be shown [1] that

$$\tan \alpha = \frac{v_{xy}}{v_z}, \quad (76)$$

where, as with the slope distribution model, α is the angle between the bisector of the incident and viewing directions and the surface normal vector \mathbf{n} . Let us assume that $\tan \alpha_o = 2 \sigma_h/T$. Then, we can write:

$$\exp \left(-\frac{v_{xy}^2 T^2}{4 v_z^2 \sigma_h^2} \right) = \exp \left(-\frac{\tan^2 \alpha}{\tan^2 \alpha_o} \right). \quad (77)$$

Many rough surfaces are gently varying, and the slopes (α) of most facets are small. Therefore, we may approximate the tangents in equation 77 by their arguments, obtaining:

$$\exp \left(-\frac{v_{xy}^2 T^2}{4 v_z^2 \sigma_h^2} \right) = \exp \left(-\frac{\alpha^2}{2 \left(\alpha_o / \sqrt{2} \right)^2} \right). \quad (78)$$

From equations 78 and 75, we see that the roughness parameters of the Torrance-Sparrow model and the Beckmann-Spizzichino model may be related as:

$$\sigma_\alpha = \frac{\alpha_o}{\sqrt{2}} = \frac{1}{\sqrt{2}} \tan^{-1} \frac{2 \sigma_h}{T}. \quad (79)$$

Figure 23 shows radiance diagrams plotted for surfaces with different roughness values using the Beckmann-Spizzichino model (left column) and the Torrance-Sparrow model (right column). Here again, only the specular lobe component is considered. Note that these radiance diagrams differ from all of the previous ones in that radiance is plotted as a function of the source angle θ_i for fixed values of the viewing angle θ_r , rather than vice-versa. Once again we assume that $\phi_r = 0$, the geometrical attenuation factor equals unity, and the Fresnel reflection coefficient is constant. For each σ_h/T ratio in the left column, we have used equation 79 to find σ_α for the corresponding diagram in the right column. Three important observations can be made from these radiance diagrams:

- When the source direction, viewer direction, and surface normal are coplanar, the radiance curves can be represented by Gaussian functions. This statement can be proved analytically by setting $\phi_r = 0$ in the specular lobe component of both models.
- The peak for each radiance curve is observed at the specular angle, i.e. $\theta_i = \theta_r$. Varying source direction, rather than viewing direction, prevents off-specular peaks from occurring. In addition, the radiance value exhibits reflection symmetry with respect to the viewer-normal plane.

- The radiance diagrams predicted by the physical optics and the geometrical optics models resemble each other very strongly. Therefore, even though the two models use two different surface modeling parameters (height and slope, respectively), equation 79 does very well in relating the physical roughness parameters of the two models.

We can further illustrate the difference between varying source direction and varying viewer direction by introducing a new representation of the reflection components. Figure 24 shows polar plots of the diffuse lobe, specular lobe, and specular spike.

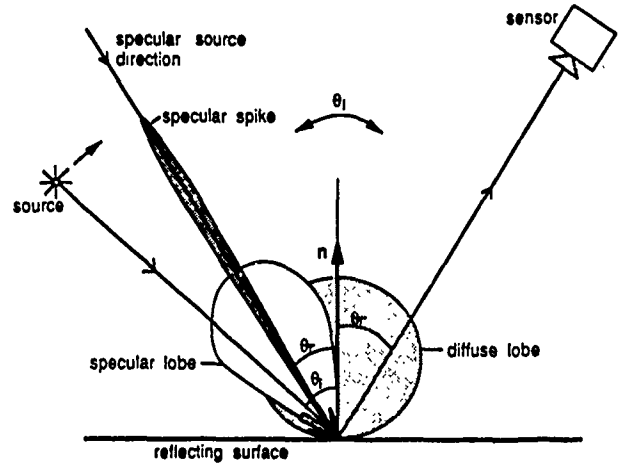


Figure 24: Polar plots of the three reflection components as functions of the source angle for a fixed viewing direction.

This time, however, the magnitudes of the three components of the radiance value in the viewing direction are determined by intersections made by the lobes with the line joining the source and the origin. In this case, the diffuse component varies with the position of the source, since it is proportional to the surface irradiance. Note that the specular lobe is symmetric with respect to the source specular angle $\theta_i = \theta_r$, and the spike is concentrated around the same angle. From the above observations, the image irradiance equation, for fixed sensor direction and varying source direction, may be written:

$$I_{im} = K_{dl} \cos \theta_i + K_{sl} \exp \left(-\frac{\alpha^2}{2 \sigma_\alpha^2} \right) + K_{ss} \delta(\theta_i - \theta_r) \delta(\phi_r) \quad (80)$$

where the constants K_{dl} , K_{sl} , and K_{ss} represent the strengths of the diffuse lobe, specular lobe, and specular spike components, respectively. Note that the ratio K_{sl}/K_{ss} is dependent on the surface roughness and the angles of incidence and reflection. Seldom are K_{sl} and K_{ss} comparable to one another.

In most instances, one of the two specular components is significant, while the other is negligible.

6 Concluding Remarks

- We propose a reflection model with three primary components: the diffuse lobe component, the specular lobe component, and the specular spike component.
- The Lambertian model may be used to represent the diffuse lobe component. This model has been used extensively to test shape-from-shading and photometric stereo techniques, and the results have indicated that it performs reasonably well. More accurate models [14] [22] may be used at the cost of functional complexity.
- The Beckmann-Spizzichino physical optics model predicts both the specular lobe and spike components. For a very smooth surface ($\sigma_h \ll \lambda$), the spike component dominates and the surface behaves like a mirror. As the roughness increases, however, the spike component shrinks rapidly, and the lobe component begins to dominate. The two components are simultaneously significant for only a small range of roughness values.
- A sharp specular component may result from the specular spike component when the surface is smooth ($\sigma_h/\lambda < 1.5$), and/or from the specular lobe component when the surface is gently undulating ($\sigma_h/T < 0.02$).
- The Torrance-Sparrow geometrical optics model provides a good approximation to the specular lobe component of the Beckmann-Spizzichino model. Both models are successful in predicting off-specular peaks in the specular lobe component. Owing to its simpler mathematical form, the Torrance-Sparrow model may be used to represent the specular lobe component.
- The Torrance-Sparrow model is not capable of describing the mirror-like behavior of smooth surfaces, and it should not be used to represent the specular spike component as it would produce erroneous results.
- The specular lobes of both Torrance-Sparrow, and Beckmann-Spizzichino models tend to have specular peaks, rather than off-specular peaks, when the viewing direction is fixed and the source direction is varied.
- Though the two models were derived using different surface modeling approaches, their surface roughness parameters may be related to one another by comparing the equations that describe their specular lobe components.

Acknowledgements

The authors are grateful to Prof. Frank Tabakin of the University of Pittsburgh for his valuable comments and guidance. The members of the VASC center at Carnegie Mellon University provided many useful suggestions. Shree K. Nayar was supported by Westinghouse Electric Corporation. This research was supported in part by Westinghouse Electric Corporation and in part by DARPA under contract F33615-87-C-1499.

References

- [1] P. Beckmann and A. Spizzichino, *The Scattering of Electromagnetic Waves from Rough Surfaces*, Pergamon Press, 1963.
- [2] E. V. Bohn, *Introduction to Electromagnetic Fields and Waves*, Addison-Wesley Publishing Co., 1967.
- [3] S. Chandrasekar, *Radiative Transfer*, Dover Publications Inc., 1960.
- [4] E. N. Coleman and R. Jain, *Obtaining 3-dimensional shape of textured and specular surface using four-source photometry*, Computer Graphics and Image Processing, Vol. 18, No. 4, pp. 309-328, April, 1982.
- [5] R. L. Cook and K. E. Torrance, *A Reflectance Model for Computer Graphics*, Proceedings, Siggraph, Vol. 15, No. 3, pp. 307-316.
- [6] G. Healey and T. O. Binford, *Local Shape from Specularity*, Proc. Image Understanding Workshop, Vol. 2, pp. 874-887, February, 1987.
- [7] B. K. P. Horn, *Hill Shading and the Reflectance Map*, Proc. of the IEEE, Vol. 69, No. 1, pp. 14-47, January 1981.
- [8] B. K. P. Horn and R. W. Sjöberg, *Calculating the reflectance map*, Applied Optics, Vol. 18, No. 11, pp. 1770-1779, June 1979.
- [9] B. K. P. Horn, *Image intensity understanding*, Artificial Intelligence, Vol. 8, No. 2, 1977.
- [10] B. K. P. Horn, *Robot Vision*, MIT Press, 1986.
- [11] A. F. Houghens and R. G. Hering, *Bidirectional Reflectance of Rough Metal Surfaces*, Thermophysics of Spacecraft and Planetary Bodies, G. B. Heller, Eds. Academic Press, 1967.
- [12] K. Ikeuchi, *Determining surface orientations of specular surfaces by using the photometric stereo method*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 3, No. 6, pp. 661-669, November, 1981.
- [13] G. J. Klinker, S. A. Shafer, and T. Kanade, *The Measurement of Highlights in Color Images*, International Journal of Computer Vision, Vol. 2, No. 1, Spring, 1988.

- [14] P. Kubelka and F. Munk, *Ein Beitrag zur Optik der Farbanstriche*, Z. tech. Physik, Vol. 12, 593, 1931.
- [15] J. H. Lambert, *Photometria sive de mensura de gratibus luminis, colorum et umbrae*, Eberhard Klett, Augsburg, 1760.
- [16] S. K. Nayar, K. Ikeuchi, T. Kanade, *Extracting Shape and Reflectance of Hybrid Surfaces by Photometric Sampling*, Proceedings of Image Understanding Workshop, Palo Alto, May 1989.
- [17] S. K. Nayar, K. Ikeuchi, T. Kanade, *Surface Reflection: Physical and Geometrical Perspectives*, CMU-RI-TR-89-7, March, 1989.
- [18] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis, *Geometrical Considerations and Nomenclature for Reflectance*, NBS Monograph 160, National Bureau of Standards, October 1977.
- [19] A. P. Pentland, *Local Shading Analysis*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 6, No. 2, pp. 170-187, March, 1984.
- [20] B. Phong, *Illumination for Computer Generated Pictures*, Communications of ACM, Vol. 18, pp. 311-317, 1975.
- [21] *Physics Handbook*, American Institute of Physics Handbook, McGraw-Hill, 1972.
- [22] J. Reichman, *Determination of Absorption and Scattering Coefficients for Nonhomogeneous Media. 1: Theory*, Applied Optics, Vol. 12, No. 8, pp. 1811-1815, August, 1973.
- [23] J. Reichman, *Determination of Absorption and Scattering Coefficients for Nonhomogeneous Media. 1: Theory*, Applied Optics, Vol. 12, No. 8, pp. 1816-1823, August, 1973.
- [24] W. A. Rense, *Polarization Studies of Light Diffusely Reflected from Ground and Etched Glass Surfaces*, Journal of Optical Society of America, Vol. 40, No. 1, pp. 55-59, January 1950.
- [25] A. C. Sanderson, L. E. Weiss, and S. K. Nayar, *Structured Highlight Inspection of specular surfaces*, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 10, No. 1, pp. 44-55, January, 1988.
- [26] R. Siegel and J. R. Howell, *Thermal Radiation Heat Transfer*, McGraw-Hill, 1972.
- [27] W. M. Silver, *Determining Shape and Reflectance Using Multiple Images*, S. M. Thesis, Dept. of Electrical Engineering and Computer Science, MIT, Cambridge, Massachusetts, June, 1980.
- [28] B. G. Smith, *Geometrical Shadowing of a Random Rough Surface*, IEEE Trans. Ant. and Propagation, Vol. 15, No. 5, pp. 668-671, Sept. 1967.
- [29] E. Sparrow and R. Cess, *Radiation Heat Transfer*, McGraw-Hill, 1978.
- [30] H. D. Tagare and Rui J. P. deFigueiredo, *A Framework for the Construction of General Reflectance Maps for Machine Vision*, Technical Report, Dept. of Elec. and Comp. Engg., Rice University.
- [31] K. Torrance and E. Sparrow, *Theory for Off-Specular Reflection from Roughened Surfaces*, Journal of the Optical Society of America, No. 57, pp. 1105-1114, 1967.
- [32] L. B. Wolff, *Spectral and Polarization Stereo Methods using a Single Light Source* Proc. Image Understanding Workshop, Vol. 2, pp. 810-820, February, 1987.
- [33] R. J. Woodham, *Photometric stereo: A reflectance map technique for determining surface orientation from image intensity*, Proc. SPIE, Vol. 155, pp. 136-143, 1978.

Local Spatial Frequency Analysis for Computer Vision

John Krumm and Steven A. Shafer *

Robotics Institute
Carnegie Mellon
Pittsburgh, PA 15213

Abstract

A sense of vision is a prerequisite for a robot to function in an unstructured environment. However, real-world scenes contain many interacting phenomena that lead to complex images which are difficult to interpret automatically. Typical computer vision research proceeds by analyzing various effects in isolation (e.g. shading, texture, stereo, defocus), usually on images devoid of realistic complicating factors. This leads to specialized algorithms which fail on real-world images. Part of this failure is due to the dichotomy of useful representations for these phenomena. Some effects are best described in the spatial domain, while others are more naturally expressed in frequency. In order to resolve this dichotomy, we present the combined space/frequency representation which, for each point in an image, shows the spatial frequencies at that point. Within this common representation, we develop a set of simple, natural theories describing phenomena such as texture, shape, aliasing and lens parameters. We show how these theories lead to algorithms for shape from texture and for dealiasing image data. The space/frequency representation should be a key aid in untangling the complex interaction of phenomena in images, allowing automatic understanding of real-world scenes.

1 Introduction

In order to function in the real world, robots need to be able to perceive what is around them through a visual sense. Unfortunately, the world is very complex, and current approaches to machine vision have not proven successful at dealing with this complexity. Because of this, most "real systems" for machine vision are actually based on many very specialized assumptions about the world: on the other hand, researchers

*This research was supported by the Defense Advanced Research Projects Agency, DoD, through ARPA Order Number 4976, monitored by the Air Force Avionics Laboratory under Contract F33615-87-C-1499 and by the Jet Propulsion Laboratory, California Institute of Technology, sponsored by the National Aeronautics and Space Administration under Contract 957989 and by the National Aeronautics and Space Administration under the Graduate Student Researcher's Program, Goddard Space Flight Center, for the first author, grant NGT-50423. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the United States Government or the Jet Propulsion Laboratory.

doing theoretical work study just one simple phenomenon at a time, but cannot deal with the interactions that are always present in realistic scenarios. These circumstances have led to very slow progress in developing real vision systems that have generality and a sound theoretical foundation.

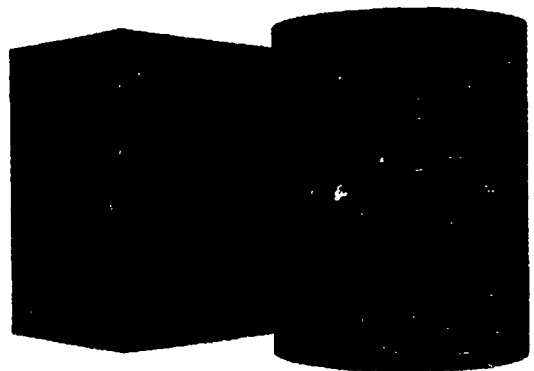


Figure 1: Cylinder and cube with Brodatz textures

In this paper, we examine the area of *spatial vision* – all of the 2D and 3D geometric factors that combine to result in the arrangement of features in the image. The factors of spatial vision include:

2D Texture Patterns "painted" on a flat, smooth surface show up as patterns in the image.

3D Texture Roughness and topography of the surface interact with lighting to produce additional patterns in the image.

Surface Shape and Perspective The 3D orientation of a surface causes its patterns to project in a particular way onto the image plane.

Image Resolution The resolution of the sensor induces sampling and aliasing in the image data, sometimes even causing noticeable moire patterns.

Focus The optics of the lens induces blurring in the imaging process due to defocus.

Other factors There are numerous other factors we shall not address further in this paper, including some whose magnitude is much smaller than the factors listed above (e.g.

diffraction), and some that involve additional imaging parameters (e.g. shadows, motion blur).

For each of the above phenomena, there has already been substantial theoretical vision research and sometimes real systems. However, the theories invariably deal with just one or just two of the above factors; and the real systems work by virtue of the highly limiting assumptions that are embedded within the algorithms, such as building in a specific size range of textures to be analyzed.

The real world is not so well-behaved. Real images exhibit these factors simultaneously, as we illustrate in Figure 1. This image, synthetically generated, shows two objects with Brodatz [Brodatz, 1966] textures mapped onto their surfaces. The textures themselves would pose a difficult analysis problem even if they were viewed frontally, as is usually presumed in research into 2D texture analysis. However, in this scene, the textures are mapped onto 3D surfaces, one curved and one polyhedral. Thus, the size and spatial relationships among the repetitive elements may change across an object or a surface. Because the resolution of the imaging sensor is finite, the texture elements or their component features may even become so small that they are blurred out of perceptibility – yet the same texture persists in that place in the real world, even though we can't explicitly see and measure it. The texture patterns themselves are not perfectly repetitive and may vary, and these variations should not be confused with the other sources of variation across a surface. And, this figure doesn't even demonstrate the effects of 3D texture – we mapped the Brodatz intensity patterns onto simulated smooth surfaces – or of defocus, which would cause the texture to blur selectively at some places in the image.

Analyzing such combinations of spatial features is far beyond the capability of current robot vision systems. Yet, the real world presents just such interactions, not just on rare occasions, but on virtually every surface in every image that we care to analyze. In order to build reliable, general vision systems, we need to explicitly understand, model, and analyze each of these phenomena and their interactions.

One of the principal reasons for the slow progress in this direction is the lack of even a suitable representation that would allow us to model all of these spatial phenomena in one framework. The use of a single framework is critical, because if each phenomena is described in a different formalism, then their interactions become combinatorially complex even to describe mathematically. But, if a single framework is used, then all of the interactions can be naturally expressed within the same vocabulary.

What framework can be used? The spatial/geometry domain provides elegant descriptions of surface shape and perspective, not-so-elegant descriptions of focus and resolution, and, as the 2D texture community has shown, poor descriptions of 2D texture and repetition. The Fourier domain appears elegant for 2D texture, focus, and resolution. Unfortunately, the frequency domain has great problems with 3D surface shape, multiple surfaces in the scene, and curved surfaces or other sources of local texture variation, because the Fourier transform mixes together frequency information from all across the image without any notion of *locality*. Obviously, no representation can be a general basis for spatial vision if it has no concept of locality within the image.

What we seek is a representation for image data that provides frequency data, but does so within the context of surfaces and other local neighborhoods of the image. There exists a class of representations that does just this: the so-called *space/frequency* distributions. These have been proposed specifically for analysis of 2D textures on flat surfaces in the past, but as shown above, that is a small part of the total

problem of spatial vision. In this paper, we show that this same class of representations can be used as an elegant representation for all of the phenomena described above, in 3D as well as 2D. We concentrate on a particular space/frequency distribution, the *image spectrogram*, because it has properties that appear most desirable for general robot vision.

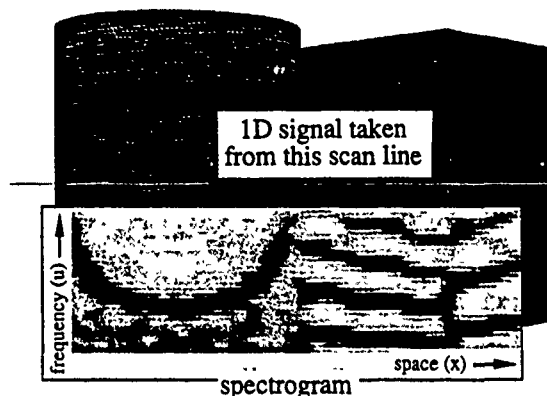


Figure 2: Figure 1 with spectrogram of center row

We show the spectrogram of the center scan-line of Figure 1 superimposed in Figure 2. The spectrogram is a two-dimensional function of space (horizontal axis) and frequency (vertical axis). Because the underlying patterns on the two objects are periodic, there are dark, frequency peaks in the spectrogram where the objects occur. The large, "U"-shaped frequency peak on the left shows that the frequency of the texture pattern projected from the cylinder appears higher near the edges than in the middle, as one would expect. At the extreme edges of the cylinder, the projected frequency is so high it cannot be adequately reproduced in the image. This is shown in the spectrogram as the frequency peak bumping into the Nyquist frequency at the top. On the left side of the cube, we see a slowly decreasing fundamental frequency and overtones which are likewise decreasing. This decrease continues to the corner of the cube, where the fundamental and harmonics begin to increase as the side recedes into the distance. This is a sample of the kind of analysis possible with the spectrogram.

The remainder of this paper explores in more detail the connections between the image spectrogram and the 3D scene. We foresee space/frequency representations as an important, unifying framework for future work in computer vision. Our research is in its early stages, so the power of the representation remains speculative but promising.

1.1 Previous Work

The work presented here draws on two separate efforts in computer vision – image representation and texture analysis. Because the space/frequency representation is so well-suited to texture analysis, researchers in that field have often used local spatial frequency decompositions for their work. Researchers in image representation have been similarly drawn to the idea, because it offers so much promise for a wide variety of analysis. However, neither effort has shown the general utility of the representation. Work in texture has not addressed the use of space/frequency representation in a formal sense, instead using the concept in an *ad hoc*, shallow manner. And

while work in image representation has addressed the concept more deeply, it is primarily concerned with issues of computation and completeness rather than actually solving problems. Our work is intended to give a deeper understanding of the representation for computer vision problems, especially for three-dimensional texture analysis.

1.2 Image Representation

There are many different choices for computing representations which combine space and frequency. One of the oldest and simplest is the spectrogram, long used on 1D signals in speech analysis. Spectrograms of 2D images were probably first used for texture segmentation of aerial images. The spectrogram is satisfactory for signals which are locally stationary (constant frequency), but tend to smear the frequency peaks of signals whose frequency is changing quickly. A similar, but more sophisticated, method of computing instantaneous frequency distributions is the Wigner distribution (WD), introduced by Wigner for use in quantum mechanics. Like the spectrogram, the WD produces a function of both space and frequency from a function of space alone.¹ An informative introduction to the WD can be found in a three-part series by Claasen and Mecklenbrauker [Claasen and Mecklenbrauker, 1980a] [Claasen and Mecklenbrauker, 1980b] [Claasen and Mecklenbrauker, 1980c]. Practically speaking, the WD can effectively deal with signals whose frequency is changing, giving a clear indication of their instantaneous frequency. Both the spectrogram and WD are joint representations of space and spatial frequency. Such image representations are reviewed and compared by Jacobson and Wechsler [Jacobson and Wechsler, 1988]. The Wigner distribution has proven unsuitable for the work described in this paper because it produces *cross terms* – peaks in the space/frequency function that do not represent any actual frequencies in the signal. The work that does use the Wigner distribution successfully is based on more global properties of the distribution rather than a detailed analysis of the distribution at each point, so cross terms are more tolerable.

An early effort aimed at creating a space/frequency representation was that of Gabor [Gabor, 1946], who proposed the use of one-dimensional, Gaussian-modulated sinusoids as basis functions that are maximally compact in both time (space) and frequency. Marčelja [Marcelja, 1980] found that these functions describe the response of visual cortex cells. The theory was extended to two dimensions by Daugman [Daugman, 1985], who showed that the two-dimensional Gabor functions can describe the cells of the visual cortex. Most work in image analysis of this type uses the Gabor functions as convolution filters, but not as a form of complete image representation. The Gabor functions are a complete, but not orthogonal, set of basis functions. Nonorthogonal basis functions complicate the process of decomposition, although it has been achieved with a neural network by Daugman [Daugman, 1988].

There has been much work on "image pyramids", starting with that of Tanimoto and Pavlidis [Tanimoto and Pavlidis, 1975]. A pyramid representation of an image consists of a series of versions of the image at ever decreasing spatial resolution. The degree of resolution at any level implicitly enforces an upper frequency limit. This is similar to the space/frequency representation in that each level of the pyramid contains only a certain band of frequencies.

Mallat [Mallat, 1989] has developed a theory for the multiresolution representation of images called an "orthogonal

wavelet representation". It is composed of a low resolution image and successively higher resolution "difference" images that fill in the details of the previous images. The representation falls between the space and frequency domains, and gives an idea of the predominant frequencies at every point in the image. A significant difference between the wavelet and Gabor representations is that the wavelet representation has orthogonal basis functions, making the representation easy to compute.

1.3 Texture Analysis

Because local spatial frequency analysis is especially well-suited to investigating repetitive patterns, the field of texture analysis has made use of this representation. There is a large set of work on texture analysis in general, so much so that at least three survey papers have been published on the topic [Haralick, 1979] [Wechsler, 1980] [Van Gool *et al.*, 1985]. This section will be restricted to comments on those efforts in which local spatial frequency analysis plays a dominant role.

There have been many efforts aimed at 2D texture segmentation using windowed Fourier transforms, for instance the work of Gramenopoulos [Gramenopoulos, 1973] and Kirvida [Kirvida, 1976]. These algorithms usually proceed by picking some set of features from Fourier space and then clustering using traditional pattern recognition techniques. The method has been compared to others both empirically by Weszka *et al.* [Weszka *et al.*, 1976] [Dyer and Rosenfeld, 1976] and theoretically by Connors and Harlow [Connors and Harlow, 1980]. While the Fourier features performed adequately, they were outperformed by other statistical texture measures. More recently, the Wigner distribution has been used for texture segmentation. Compared to windowed Fourier transforms, the WD can more effectively deal with signals whose frequency is changing, giving a clear indication of their instantaneous frequency. It has been applied to texture segmentation by Reed and Wechsler [Reed and Wechsler, 1990]. Gabor-function filtering has been applied to the tasks of texture segmentation by Turner [Turner, 1986] and Bovik *et al.* [Bovik *et al.*, 1990]. Fogel and Sagi [Fogel and Sagi, 1989] found that Gabor function texture segmentation closely paralleled human performance. All of this work in 2D texture, however, has used the frequency data as merely features to be grouped rather than trying to attach any higher-level meaning as we do.

One form of higher-level meaning is the effect of three-dimensional effects on image texture. This was first investigated by Bajcsy and Lieberman [Bajcsy and Lieberman, 1976] who computed texture gradients in images using non-overlapping, windowed Fourier transforms. Their insight into the problem was largely qualitative, and their methods served as demonstrations rather than useful algorithms. The only other research like this was that of Jau and Chin [Jau and Chin, 1988]. They computed slant and tilt angles of textured planes using integrated, low-frequency regions of the Wigner distribution. Although the results were good, the research did not show anything more than an empirical connection between shape and frequency, and it was based on arbitrary values for the window size and low-pass frequency limit.

Other work which is more peripherally related to texture in computer vision shows the wider, potential utility of the space/frequency representation. Heeger [Heeger, 1988] showed how Gabor function filtering of the space/time cube could be used to extract image motion of moving texture fields. Matsuyama *et al.* [Matsuyama *et al.*, 1983] used Fourier transforms taken over regions of uniformly distributed texture elements in order to find the two spatial vectors that characterize

¹Much of the work in space/frequency representations is presented in terms of time, not space.

the placement of the elements. The Fourier transform has also been considered for calculating the point of best focus for an entire image by Horn [Horn, 1968], and for a subsection by Krotkov [Krotkov, 1987]. Pentland used the Fourier transform for both shape from focus [Pentland, 1985] and shape from shading [Pentland, 1988].

Compared to the previous work in texture, our work is different in that, although it uses a similar representation, it goes farther by taking into account many other phenomena in addition to just texture. The previous texture work using spatial frequencies has not emphasized the detailed, quantitative interaction between scene parameters and frequency, and has not presented the space/frequency representation as one for general image understanding. In contrast, other researchers in space/frequency representations *have* speculated that the representation may prove useful for computer vision applications, but have not shown it to be true in general. We use an existing method of computing the space/frequency representation (the spectrogram) and demonstrate how it can be used to reason naturally about a wide variety of phenomena, emphasizing the effect of three-dimensional shape.

1.4 This Paper

In this paper we show how a joint space/frequency representation can be used to effectively examine a variety of important phenomena in computer vision. In the next section, we examine two of the most popular joint representations – the spectrogram and the Wigner distribution – and we describe which one is most useful to us. In Section 3 we show how the spectrogram maintains coherence over regions of similar texture, even if the texture is changing in frequency. Making this coherence explicit means that the spectrogram can be used for segmentation on textures other than just those on a plane viewed frontally, which is an implicit limitation in most texture segmentation algorithms. In Section 4 we show how textured shapes affect the spectrogram. We examine in detail the spectrogram of a texture along a line and demonstrate how we can accurately extract shape parameters in this simple case. Section 5 shows how spatial aliasing (moiré patterns) affects the spectrogram. In Section 6 we show how changes in a camera's lens parameters (zoom, focus, and aperture) affect the spectrogram in a predictable way. The zoom analysis, combined with the development on aliasing, leads to an algorithm for dealiasing images of simple textures. We examine other issues in Section 7.

2 Space/Frequency Representations

Contiguous texture patterns in a scene normally do not appear as constant frequency patterns in an image, because the underlying shape is usually not planar. Even if it were, the frequency would only appear constant if the texture were viewed along the plane's normal. Thus, frequency analysis of texture in nontrivial scenes requires a method which can account for changes in frequency with position. This is beyond the ability of conventional, large support, Fourier transforms, so other methods have been devised.

An illustration of the space/frequency representation is shown in Figure 3. Figure 3-a shows two sinusoidal waves in which the higher-frequency wave occupies the center quarter of the signal. The Fourier transform of this signal is shown in Figure 3-b. Although it shows two pairs of frequency peaks, it does not show *where* in space the subsignals of corresponding frequency occur. The structure of the signal is made clear in the space/frequency representation of Figure 3-c, which shows that a relatively low-frequency component exists at the ends of the signal in question, while a higher-frequency

part occurs in the middle one quarter. This localization is the power of the space/frequency representation.

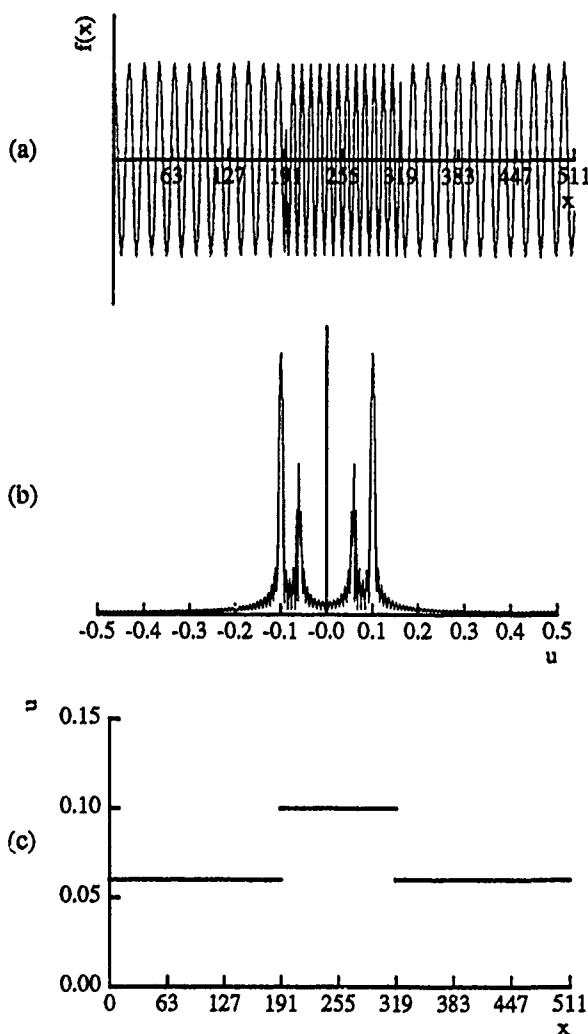


Figure 3: (a) two sinusoids (b) Fourier transform shows only frequency (c) space/frequency representation shows structure

Signals whose frequency changes with position are called *nonstationary*. A simple example is $\cos(2\pi u_0 x^2/2)$. The *instantaneous frequency* of such a signal is defined as the derivative of the argument with respect to the spatial variable – in this example, $u_0 x$ (in cycles/unit distance). Certain frequency-based, texture segmentation algorithms [Dyer and Rosenfeld, 1976] do not require an accurate estimate of the instantaneous frequency, only one which is sensitive to significant differences in frequency. Thus, they can work with only a coarse sampling in frequency. In our work, however, we are concerned with small changes in frequency, due to, for instance, surface slope or variations in zoom. Thus, we require a high resolution, accurate estimate of the instantaneous frequency.

We consider in this section the two primary means of calculating space/frequency representations: the spectrogram and the Wigner distribution. A third method is to fit sinusoids to the signal over small windows; although it is slow, it leads to high resolution estimates. Both this method and the spectrogram are based on the assumption that the signal is locally

stationary. The WD relaxes this assumption.

Our analysis in this section and the rest of this paper will be limited to one-dimensional signals. This not only simplifies understanding the mathematics, but makes visualization of the representation much easier. For a 1D signal, the space/frequency representation is two-dimensional, while for a 2D signal (an image), it is four-dimensional. Our example spectrograms are superimposed on 2D images. In these figures, the spectrogram was computed from the center row of the image. We include the entire image to illustrate more clearly the various effects we are considering.

2.1 The Spectrogram

The spectrogram of a signal is a series of small-support, Fourier transforms of the signal, each centered around a different point of the signal. For a one-dimensional signal $f(x)$, the spectrogram is $S_f(x, u)$, where u is frequency in cycles/unit distance. $S_f(x, u)$ is an estimate of the power of frequency u at the point x . The continuous spectrogram of the one-dimensional function $f(x)$ is given by

$$S_f(x, u) = \left| \int_{-\infty}^{\infty} w_l(\alpha - x) f(\alpha) e^{-j2\pi u \alpha} d\alpha \right|^2,$$

where $w_l(x)$ is a window function with support length l .

The process by which a spectrogram is calculated is shown in Figure 4. To calculate one vertical slice of the spectrogram for a given value of x , say x_0 , the signal is first multiplied by a window offset by x_0 . This product is Fourier transformed; the magnitude is calculated from the complex values of the Fourier transform; and the non-negative half of the magnitudes serve as $S_f(x_0, u)$, which is one column of the spectrogram. This process is repeated for every x . We only consider the non-negative half of the magnitudes since the Fourier transform of a real signal (the only kind we have) is symmetric in magnitude. The discrete version is computed using the discrete Fourier transform (DFT), which is discrete in both space and frequency. The window function controls how much of the rest of the signal contributes to the spectrogram at the point x . In terms of $W_l(u)$ and $F(u)$, the Fourier transforms of $w_l(x)$ and $f(x)$, the spectrogram is

$$S_f(x, u) = \left| (e^{-j2\pi x u} W_l(u)) * F(u) \right|^2, \quad (1)$$

where "*" is convolution.

The spectrogram of a two-dimensional function $f(x, y)$ is a straightforward extension of the equation above, giving a four-dimensional spectrogram, $S_f(x, y, u, v)$, with two spatial variables and two frequency variables.

There are ongoing questions about the best shape and size of the window $w_l(x)$. Many window shapes are considered by Harris in [Harris, 1978]. He illustrates the compromises involved in the selection, and concludes by recommending the 4-sample Blackman-Harris window. We use the minimum, 4-sample Blackman-Harris window, which for a discrete set of n points is given by

$$w_n(k) = a_0 - a_1 \cos\left(\frac{2\pi}{n-1}k\right) + a_2 \cos\left(\frac{2\pi}{n-1}2k\right) - a_3 \cos\left(\frac{2\pi}{n-1}3k\right) \quad (2)$$

for $k = 0, 1, \dots, n-1$ and $(a_0, a_1, a_2, a_3) = (0.35875, 0.48829, 0.14128, 0.01168)$.

The window size l (or in the discrete case n) affects how much of the signal is included in the Fourier transform at each

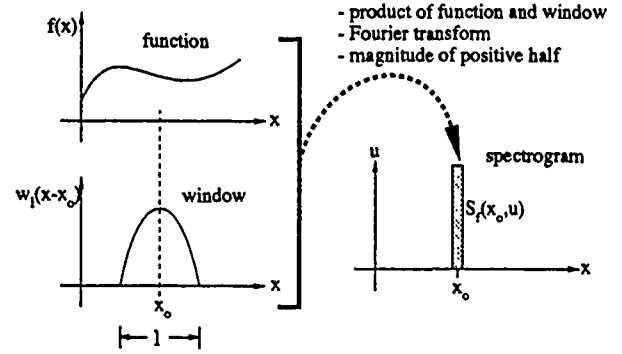


Figure 4: Computing the spectrogram

point. Equation 1 above shows that the effect of windowing is to convolve the Fourier transform of the signal, $F(u)$, with the Fourier transform of the window, $W_l(u)$. This can be thought of as a blurring of the signal's spectrum with $W_l(u)$. As the width of the window decreases, the width of W_l grows, meaning that the spectrum will be more smeared. Thus, a large window is desirable for a sharp spectrum. However, a large window will compromise the localization ability of the spectrogram, as it will include components of the signal which are distant from the point of interest. In practice, we have found $n = 63$ to be satisfactory on discrete signals of length 512 (one image scan-line). We investigate a more sophisticated windowing technique in Section 7.1.

2.2 Wigner Distribution

An alternative method of calculating a joint space/frequency representation of a signal is the Wigner distribution. The Wigner distribution has been used in the computer vision community for both texture segmentation [Reed and Wechsler, 1990] and shape from texture [Jau and Chin, 1988]. For a one-dimensional function $f(x)$, the Wigner distribution is

$$W_f(x, u) = \int_{-\infty}^{\infty} f(x + \alpha/2) f^*(x - \alpha/2) e^{-2\pi i u \alpha} d\alpha.$$

In words, the way to compute $W_f(x, u)$ is to first calculate the product $f(x + \alpha/2) f^*(x - \alpha/2)$, which is the original signal multiplied by a conjugated version of the original signal flipped around the point x . This product is Fourier transformed to get the WD at x . In practice, $f(x)$ is first windowed, leading to the pseudo-Wigner distribution (PWD) [Claassen and Mecklenbrauker, 1980a]. The open questions pertaining to the window function for the spectrogram also apply to the PWD.

The WD generally works best on analytic signals, i.e. signals whose Fourier transforms contain no negative frequencies [Boashash, 1988]. It is fairly straightforward to calculate an analytic signal which corresponds to a real signal defined by samples. Thus, our two examples will be for analytic signals.

The example to which many WD advocates point is the WD of the chirp signal $f(x) = e^{j2\pi u_0 x^2/2}$. This nonstationary, complex sinusoid is the analytic extension of $\cos(2\pi u_0 x^2/2)$, whose instantaneous frequency is $u_0 x$ (frequency proportional to x). The WD is

$$\begin{aligned}
W_f(x, u) &= \int_{-\infty}^{\infty} e^{j2\pi u_0(x+\alpha/2)^2/2} e^{j2\pi u_0(x-\alpha/2)^2/2} e^{-j2\pi u\alpha} d\alpha \\
&= \int_{-\infty}^{\infty} e^{j2\pi u_0 x \alpha} e^{-j2\pi u\alpha} d\alpha \\
&= \delta(u - u_0 x).
\end{aligned}$$

In (x, u) space, this is a δ -ridge which tracks at exactly the instantaneous frequency of $f(x)$. For any x , the position of the ridge is at $u_0 x$, which is exactly what we would like to see for this signal.

Most textures are not simple sinusoids, however. They are, rather, sums of sinusoids in the sense of Fourier series. It is desirable that the joint representation show multiple frequency peaks at the constituent frequencies of the texture. This means that the representation should be linear – that the representation of the sum of two sinusoids should be the sum of the representations of the two sinusoids by themselves. Unfortunately, the WD is not linear. That is, $W_{f+g}(x, u) \neq W_f(x, u) + W_g(x, u)$. We show in Figure 5 the spectrogram (on the left) and the Wigner distribution (on the right) of a sum of two sinusoids. Let $f(x) = e^{j2\pi u_f x}$ and $g(x) = e^{j2\pi u_g x}$, both constant-frequency, complex sinusoids with frequencies u_f and u_g respectively. We have

$$\begin{aligned}
W_f(x, u) &= \delta(u - u_f), \\
W_g(x, u) &= \delta(u - u_g), \\
W_{f+g}(x, u) &= W_f(x, u) + W_g(x, u) + \\
&\quad 2 \cos[2\pi x(u_f - u_g)] \delta\left(u - \frac{u_f + u_g}{2}\right).
\end{aligned}$$

Thus the WD of a single, complex sinusoid is what we would expect, but the WD of a sum of sinusoids has a cross term. This term is a δ in u at the mean frequency of the two original sinusoids, modulated in x at a frequency which is the difference in frequencies of the two original sinusoids. The WD gives cross terms for every pair of constituent sinusoids. The cross term of the WD is clearly visible in Figure 5.

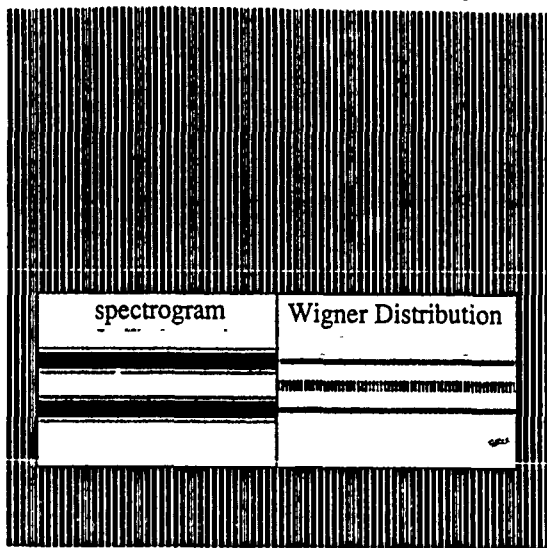


Figure 5: Spectrogram and Wigner distribution of two summed sinusoids

The analysis that follows in this paper depends on accurately finding the frequency peaks in the joint representation. Noise in some of the images complicates this task. The cross

terms introduced by the WD would make it even more difficult to distinguish the true frequency peaks. It is for this reason that we have chosen not to use the Wigner distribution.

The WD is just one member of a more general family of joint representations. Others [Choi and Williams, 1989][Zhao *et al.*, 1990], may be able to deal with nonstationarities as well as the WD while still suppressing cross terms. However, there does not exist a definitive method for calculating the space/frequency distribution.

3 Two-Dimensional Texture Segmentation With the Spectrogram

It is often the case that regions in an image can be grouped by their similarities in texture. In segmenting a road image, for example, it may be that the only common feature that the grassy areas share is texture, because the intensity and color of the grass in the image may be very different from shadowed to nonshadowed regions. By *two-dimensional* texture segmentation we mean segmentation on images with textures whose frequency does not change appreciably over the image. The textures must be viewed frontally; this is how almost all texture segmentation algorithms are tested.

The spectrogram of a structured texture shows that the spectrogram gives a clear, easily interpretable representation of the texture and a good idea of the texture's boundaries. In Figures 6 and 7 we present two pairs of textures along with the spectrograms of the rows indicated by the lines across the middle of the images. The smaller, left plate in Figure 6 has a sinusoidal intensity pattern, while the larger plate visible on the right has a square wave pattern. The left half of the spectrogram shows one peak in frequency which is constant with respect to position, as we expect from a sinusoidal intensity pattern. The right half of the spectrogram shows the fundamental frequency of the square wave pattern as the dark line near the bottom of the spectrogram along with fainter overtones at evenly spaced intervals above. The frequency of the square wave's first harmonic happens to be about equal to the frequency of the sinusoid on the left. The sharp transition between the two textures produces a short region in the spectrogram where nearly all frequencies are present. The light, vertical bars on the right half of the spectrogram are due to the interaction of the simulated pixels with the periodic pattern.

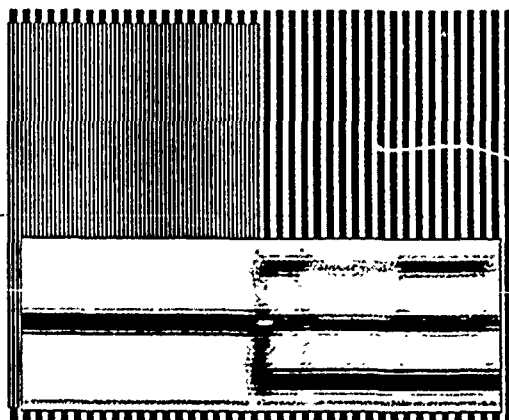


Figure 6: Two plates with sinusoidal and square wave gratings

Figure 7 shows the same two plates with Brodatz textures

superimposed. The complexity of the Brodatz images makes the spectrograms messier, but the representation is still easy to interpret. The white band at the bottom of the spectrogram has been zeroed to eliminate low frequency intensity variations due to lighting. We see that the scan line of the canvas texture on the left is close to sinusoidal since it has only one significant frequency component. The screen texture on the right has a lower fundamental frequency than the canvas as well as some overtones.

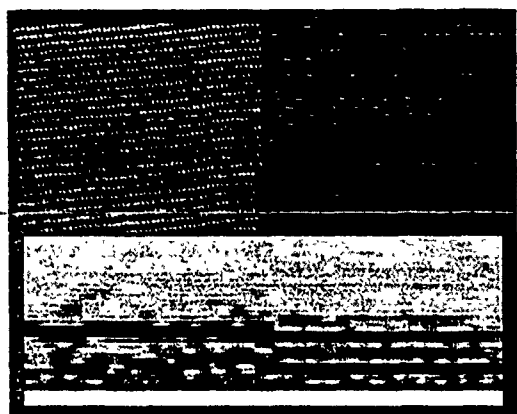


Figure 7: Two plates with Brodatz textures

There have been many efforts aimed at 2D texture segmentation using windowed Fourier transforms, for instance [Gramenopoulos, 1973] and [Kirvida, 1976]. These algorithms usually proceed by picking some set of features from Fourier space and then clustering using traditional pattern recognition techniques. The method has been compared to others both empirically [Weszka *et al.*, 1976][Dyer and Rosenfeld, 1976] and theoretically [Conners and Harlow, 1980]. While the Fourier features performed adequately, they were outperformed by other statistical texture measures.

The advantages of Fourier texture segmentation come from the variety of textures it can manage and the ease with which it can be extended to textures which are viewed obliquely. For structural textures, the Fourier transform approach requires no feature detection. Windowed Fourier transforms can be used for purely statistical textures, because Fourier transforms can bring out statistical coherence. In all textures, the spectra remain coherent over changes in shape, as we show in the next section, which means that the method can be smoothly extended to non-frontally viewed textures. In addition, the spectrogram is a powerful framework for analyzing many other scene phenomena (as we will show) and can be used to extract intrinsic scene characteristics. These intrinsic parameters provide another, more reliable basis for segmentation (Section 4.2).

4 Three-Dimensional Shape and the Spectrogram

Texture is an important indication of 3D shape, and the connection has been studied extensively in computer vision [Kender, 1980][Stevens, 1981][Witkin, 1981]. The projected, local spatial frequencies on a textured surface change with the surface's depth and orientation. This is the phenomenon which makes shape from texture possible, and it is

the reason that the spectrogram is a natural choice for this kind of analysis. In Figure 8 we show a plate receding into the distance with a sinusoidal intensity pattern superimposed. The spectrogram of the center scan line shows that the projected frequency increases as the plate recedes. This scene was contrived to show the effect of a *vanishing line*. The plate asymptotically approaches a point in the image, while the frequency peak in the spectrogram asymptotically approaches a line which corresponds to the plate's vanishing point. Before the plate reaches the vanishing line in the image, the frequency has so grown that the spectrogram shows aliasing (see Section 5).

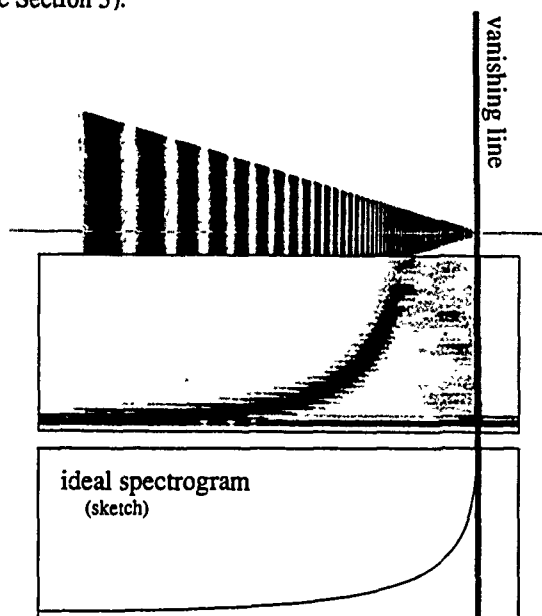


Figure 8: Plate with sinusoid receding to vanishing point

Figure 9 shows two plates meeting at a convex corner, each with a sinusoidal intensity pattern. The spectrogram shows how the projected pattern increases in frequency as the plates recede.

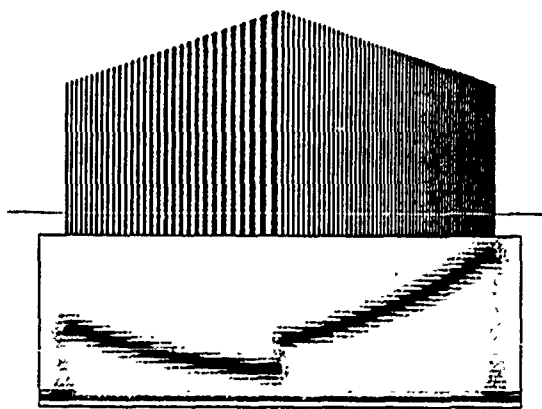


Figure 9: Two plates with sinusoids forming a convex corner

In Figures 10 and 11 we show the plates of Figures 6 and 7 rotated around a vertical axis. Both the fundamental frequencies and the overtones show the same reaction to the change

in orientation. In the following discussion, we describe how to quantitatively extract shape information from the spectrograms of textured surfaces by calculating the effect of depth and orientation on the spatial frequencies of the texture pattern.

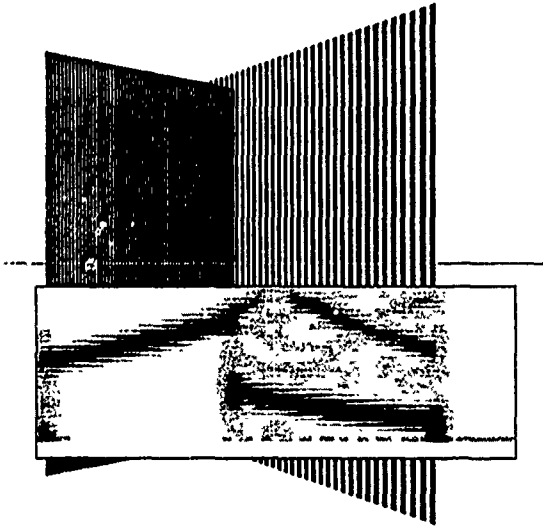


Figure 10: Two rotated plates with sinusoidal and square wave gratings

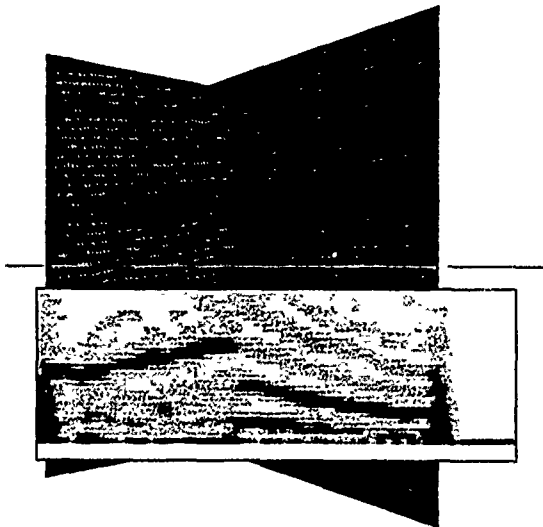


Figure 11: Two rotated plates with Brodatz textures

4.1 Mathematical Formulation

The coordinate system and other quantities are defined as in Figure 12. The pinhole of a pinhole camera is placed at the origin of the right-handed (x_{3D}, y_{3D}, z_{3D}) coordinate system, looking along the $-z_{3D}$ axis. Objects are projected onto the image whose axes are (x, y) . The pinhole-to-sensor distance is d , meaning that point (x_{3D}, y_{3D}, z_{3D}) will be projected onto the image plane at the point $(x, y) = (\frac{z_{3D}d}{-z_{3D}}, \frac{y_{3D}d}{-z_{3D}})$ under perspective. There is a surface in front of the camera whose depth is given by the function $c(x_{3D}, y_{3D})$. Superimposed on the surface is an intensity pattern given by $g(s, t)$, where (s, t) are coordinates of a coordinate system on the surface. We will ignore the y_{3D} and y coordinates, in effect confining our

attention to the x_{3D} - z_{3D} plane ($y_{3D} = 0$) and a 1D image plane in x .

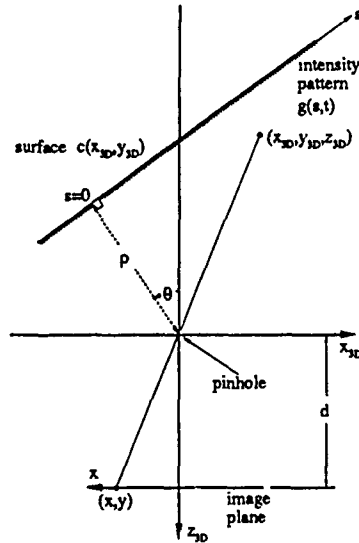


Figure 12: Geometry of 1D image formation through pinhole

On the x_{3D} - z_{3D} plane, a line runs in front of the camera whose equation is $x_{3D} \sin \theta + z_{3D} \cos \theta = -\rho$.² We will suppose that this line has a periodic pattern superimposed on it. We will find the perspective projection of this pattern onto the image plane, and then calculate the instantaneous frequency of the projection so we can apply the spectrogram. We will find that the instantaneous frequency is a function of the orientation of the line, meaning that the spectrogram can be used to determine this parameter. Points on this line are parameterized by s , where $s = 0$ occurs at the intersection of the line and its perpendicular to the origin. Given an s , we have

$$(x_{3D}, z_{3D}) = (-\rho \sin \theta + s \cos \theta, -\rho \cos \theta - s \sin \theta)$$

which projects to

$$x = d \frac{-\rho \sin \theta + s \cos \theta}{\rho \cos \theta + s \sin \theta}.$$

Solving for s , we have the position along the line for a given x on the image plane:

$$s(x) = \frac{-d\rho \sin \theta - x\rho \cos \theta}{x \sin \theta - d \cos \theta}. \quad (3)$$

Suppose that the line has superimposed on it a periodic reflectance pattern given by $g(s) = \cos(2\pi u_1 s)$, such that the frequency of the pattern along the line is u_1 . If the pattern is projected onto the image plane, we can write the equation of the projected pattern by replacing the s in $\cos(2\pi u_1 s)$ with the equivalent value of s given in terms of x in Equation 3. Thus, the projected pattern on the image plane will be given by

$$\cos[2\pi u_1 s(x)] = \cos \left[-2\pi u_1 \rho \frac{d \sin \theta + x \cos \theta}{x \sin \theta - d \cos \theta} \right].$$

The instantaneous frequency, $u(x)$, of $\cos[2\pi u_1 s(x)]$ is defined in the signal processing literature to be the derivative of the argument with respect to x , which is

²In terms of traditional shape-from-texture notation (c.f. [Witkin, 1981]), the tilt angle here is always zero because we are working in only two dimensions, while θ is like the slant angle except that the slant angle cannot be negative and θ can be.

$$u(x) = \frac{u_1 \rho d}{(x \sin \theta - d \cos \theta)^2} \quad (4)$$

The peak frequency in the spectrogram of the projected cosine will occur at approximately this frequency. In a computer vision application, the known quantities in Equation 4 are d (the pinhole-to-sensor distance), x (the pixel position), and $u(x)$ (the instantaneous frequency from the spectrogram). The unknowns are u_1 (the frequency of the pattern along the line), and ρ and θ (the parameters of the line). Since u_1 and ρ occur as a product in Equation 4, they cannot be distinguished from each other. This is a manifestation of a familiar effect: a small object (high frequency) at a small distance is indistinguishable from a large object (low frequency) at a large distance. Thus, we treat the product $u_1 \rho$ as a single unknown. With θ as the other unknown, we can solve Equation 4 for θ and $u_1 \rho$ if we have two or more sets of $(x, u(x))$. The result is a space/frequency formulation of the shape-from-texture paradigm.

4.2 Extracting Shape from the Spectrogram

To demonstrate the use of Equation 4, we will determine parameters of the two plates in Figure 10 based on the spectrogram of the center row. We simplify the spectrogram to $u(x)$, the dominant frequency, determined by finding the maximum value in each column of the spectrogram. These values are shown in Figure 13 as the dotted, staircase-like line. The staircase effect is due to the limited resolution of the DFT, which is in turn due to the limited size of the window used to calculate the spectrogram. This low resolution means that many adjacent points will appear to have equal instantaneous frequencies. If the instantaneous frequency of two adjacent points is equal, it implies that the surface is perpendicular to the line of sight, which is usually not the case. Thus, we calculate a "subpixel" value of the instantaneous frequency which gives better resolution than the raw DFT. We calculate the subpixel estimate by fitting a quadratic to the peak value and its two vertical neighbors and then finding the maximum of the quadratic. This is done for each column in the spectrogram. The higher resolution estimate is shown as the solid line in Figure 13. As a point of reference, we show the actual instantaneous frequencies (calculated from Equation 4) as the dash-dot line in the same figure. The estimate based on the spectrogram seems to consistently underestimate the actual frequency, and we are currently investigating the reason.

Each pair of $(x, u(x))$ values from the high-resolution spectrogram estimates can be used to calculate a value of $(u_1 \rho, \theta)$. In order to reduce the effects of the wavering in the instantaneous frequencies, we calculate each $(u_1 \rho, \theta)$ using five pairs of $(x, u(x))$'s placed symmetrically around the point of interest. We then segment the regions by histogramming the $(u_1 \rho, \theta)$'s, manually picking the peaks, and classifying each $(u_1 \rho, \theta)$ pair by finding which peak it is closest to.

The resulting segmentation is shown in Figure 14. The bar across the middle of the image indicates the regions, and we show the dominant instantaneous frequencies below. This segmentation works not only in spite of the changing frequencies across similar regions, but *because* of the changing frequencies as dictated by the mathematical projection of a single 3D plane onto a 2D image. In contrast to traditional region-grouping methods, note that this segmentation is based on reasoning about the uniformity of intrinsic properties of the scene, not merely the uniformity of a property in the image. In this sense, it is based on the "model coherence" approach developed for color image segmentation [Shafer *et al.*, 1990].

With the regions segmented, we calculate the best fit $(u_1 \rho, \theta)$ from Equation 4 based on the region's $(x, u(x))$'s using a gra-

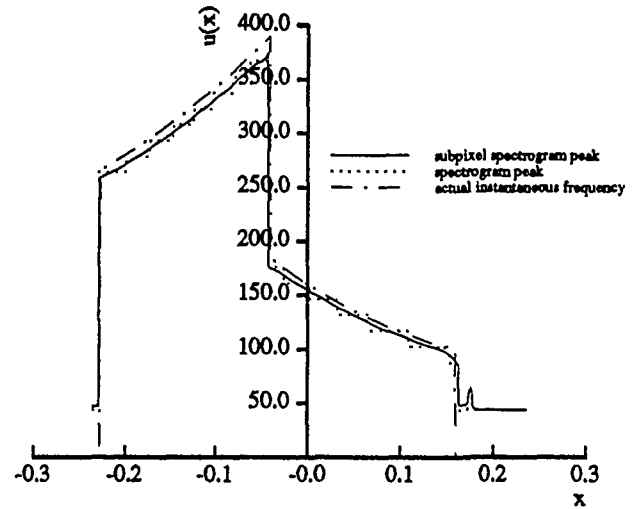


Figure 13: Peak frequencies from spectrogram of Figure 10

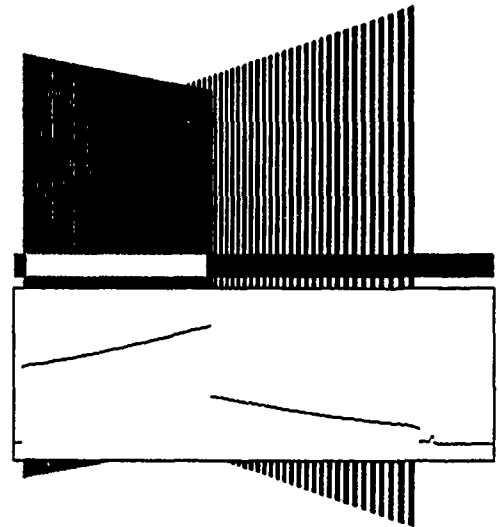


Figure 14: Segmentation of center row of rotated, patterned plates

dient descent, minimization routine. The results are shown in Table 1. We know the actual values of the parameters from the graphics routine used to generate the images. In this example the errors are quite small.

We performed the same analysis for the textured plates in Figure 11. The results of the segmentation are shown in Figure 15. This segmentation is not as good as for the other set of plates. Much of the error occurs near the boundaries of the plates where the Fourier transform window contains only part of one of the textures or some of both. The other misclassified areas occur in regions where the instantaneous frequency value has unusual dips or wiggles. Possible solutions to this problem are using a spectral estimator which accounts for noise, or averaging the dominant frequencies from the spectrograms of neighboring points. Also, using a variable-sized window as described in Section 7.1 may help alleviate the problem. The performance figures in Table 1 are based on a manual (perfect) segmentation of the instantaneous frequencies for the rotated, textured plates of Figure 11.

	From Figure 10 Periodic Pattern semi-automatic segmentation				From Figure 11 Brodatz Textures manual segmentation			
	Left Plate		Right Plate		Left Plate		Right Plate	
	$u_1\rho$	θ	$u_1\rho$	θ	$u_1\rho$	θ	$u_1\rho$	θ
actual	177.25	50.00°	40.00	-60.00°	152.1	50.00°	47.0	-60.00°
calculated	172.92	49.75°	39.31	-59.72°	141.37	50.82°	48.27	-58.85°
error	-2.4%	-0.25°	-2.4%	0.28°	-7.1%	0.82°	2.7%	1.15°

Table 1: Actual and calculated line parameters

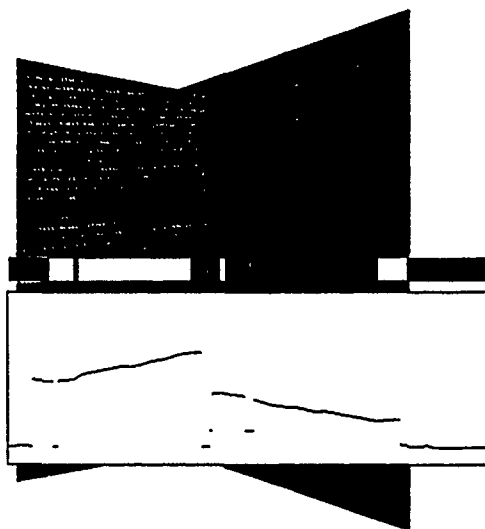


Figure 15: Segmentation of center row of rotated, textured plates

4.3 Other Shapes

This method could be extended to other shapes in two different ways. Above we presented a method in which the instantaneous frequencies are fit to a known class of shapes (lines) in order to derive the parameters of the shape. The parameters were those which best fit Equation 4, which describes the instantaneous frequencies on a line. Other equations could be derived which relate instantaneous frequencies to any parameterized shape. Given some *a priori* knowledge of the shapes in the scene, the spectrogram peaks (as well as overtones) could be used to instantiate the shapes' parameters. Alternatively, a program could calculate local surface normals by using the instantaneous frequencies from a small neighborhood along with an equation which relates frequency and surface normal.

Although this method and results are meant to be only illustrative, they show the power of the method for analyzing the effects of 3D shape in images. The spectrogram is a simple, natural method of quantifying the relationship between texture and shape, and it requires no feature detection except for finding frequency peaks.

5 Aliasing

Aliasing occurs when a signal is sampled at a rate less than twice its maximum frequency, causing lower-frequency artifacts to appear in the sampled signal. This phenomenon can often be seen on television in images of periodic patterns like striped clothes, automobile grills, or tall buildings. In two dimensional imaging, these artifacts are called *moire patterns*, and they can lead to insidious problems in machine vision,

e.g. stereo matching errors [Matthies, 1989](p. 117). This is because the patterns cannot be detected in single images without detailed *a priori* knowledge of the scene, meaning that in most situations there is no hope of recovering the true signal.

The DFT of such a signal does not give a true indication of the original signal's frequency content. The DFT can only show frequencies up to and including the Nyquist frequency (one half of the sampling frequency). Frequencies higher than the Nyquist frequency are "aliased down" into lower frequencies of the DFT.

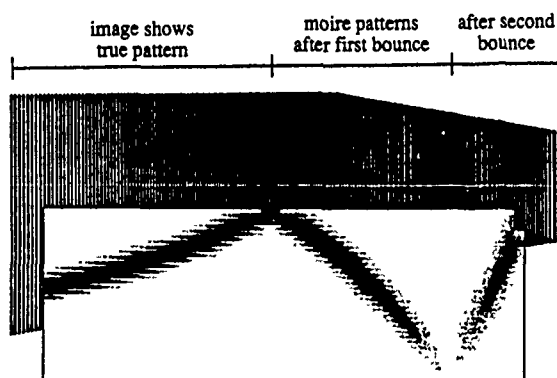


Figure 16: Plate with sinusoid showing aliasing

This is illustrated in Figure 16, which shows a plate with a sinusoidal intensity pattern rotated to the right. Beginning at the left of the plate, the spectrogram shows that the instantaneous frequency is rising as the plate recedes into the distance. At a little less than halfway across the spectrogram, the peak frequency has risen to the top of the spectrogram, which corresponds to the Nyquist frequency. Although the actual frequency on the image plane continues to rise, it appears to decrease after the Nyquist rate has been exceeded. In this region of the image, moire patterns begin to appear as lower-frequency variations caused by the beating of the signal frequency against the sampling frequency. There is another "bounce" on the spectrogram after the apparent peak frequency has fallen to zero. This bouncing would continue if the plate were longer. If the signal had overtone frequencies, these will bounce also, although not at the same places as the fundamental or other overtones. This is shown in Figure 17, which is a plate whose intensity pattern is the sum of two sinusoids. Below we examine the mathematics of the bouncing frequencies and show how the spectrogram provides an

elegant basis for analyzing these artifacts.

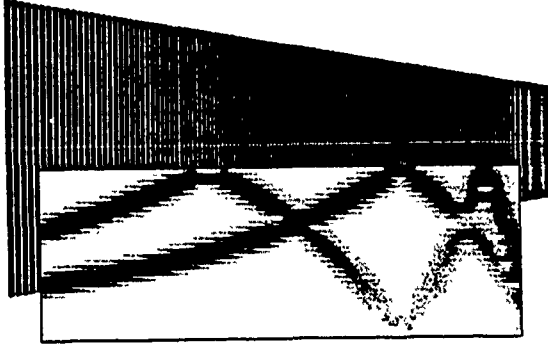


Figure 17: Plate with sum of two sinusoids showing aliasing

5.1 Bouncing Frequencies

In this section we discuss the mathematics of aliasing and how it produces bouncing in the spectrogram. We will demonstrate the effect using a simple cosine wave, although the ideas are generally applicable. The effect is most easily visualized in the Fourier domain, so we will develop the equations in the spatial and spatial frequency domains in parallel.

Suppose the original, continuous signal is a cosine of frequency u_o cycles per unit distance.

$$f(x) = \cos(2\pi u_o x)$$

Its Fourier transform is two delta functions placed symmetrically around the frequency origin.

$$F(u) = \frac{1}{2} [\delta(u + u_o) + \delta(u - u_o)]$$

Sampling at a frequency of u_s is modeled as multiplication by a series of δ 's spaced at intervals of $1/u_s$. The sampled signal, f_s , is

$$\begin{aligned} f_s(x) &= f(x) \sum_{i=-\infty}^{\infty} \delta(x - \frac{i}{u_s}) \\ &= \cos(2\pi u_o x) \sum_{i=-\infty}^{\infty} \delta(x - \frac{i}{u_s}) \end{aligned}$$

The corresponding operation in the Fourier domain is convolution with the Fourier transform of the space-domain δ 's.

$$\begin{aligned} F_s(u) &= F(u) * u_s \sum_{i=-\infty}^{\infty} \delta(u - iu_s) \\ &= \frac{1}{2} [\delta(u + u_o) + \delta(u - u_o)] * u_s \sum_{i=-\infty}^{\infty} \delta(u - iu_s) \\ &= \frac{u_s}{2} \left[\sum_{i=-\infty}^{\infty} \delta(u + u_o - iu_s) + \sum_{i=-\infty}^{\infty} \delta(u - u_o - iu_s) \right] \end{aligned}$$

$F_s(u)$, the Fourier domain version of the sampled cosine wave, is illustrated in Figure 18-a. It consists of the Fourier transform of the cosine repeated at intervals of u_s , the sampling frequency. These repeated Fourier transforms are called *spectral orders*. Spectral order $o_i \in \{\dots -2, -1, 0, 1, 2, \dots\}$ is centered at frequency $o_i u_s$.

In order to recover an estimate of the original signal from the samples, the Fourier domain representation is multiplied by a rectangle function to extract one repetition of the repeated transforms. (It is also scaled by $\frac{1}{u_s}$ to recover the original amplitude.) The rectangle function, also shown in Figure 18-a, is cut off at the positive and negative Nyquist frequencies. This corresponds to interpolation with a sinc function in the spatial domain. Thus, the reconstructed signal becomes

$$\begin{aligned} f_r(x) &= f_s(x) * \text{sinc}(u_s x) \\ &= \left[\cos(2\pi u_o x) \sum_{i=-\infty}^{\infty} \delta\left(x - \frac{i}{u_s}\right) \right] * \text{sinc}(u_s x) \\ &= \sum_{i=-\infty}^{\infty} \cos(2\pi u_o i / u_s) \text{sinc}[u_s(x - i / u_s)], \end{aligned}$$

where $\text{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$.

In the Fourier domain,

$$\begin{aligned} F_r(u) &= \frac{1}{u_s} \text{rect}\left(\frac{u}{u_s}\right) F_s(u) \\ &= \frac{1}{2} \text{rect}\left(\frac{u}{u_s}\right) \left[\sum_{i=-\infty}^{\infty} \delta(u + u_o - iu_s) + \sum_{i=-\infty}^{\infty} \delta(u - u_o - iu_s) \right], \end{aligned}$$

where

$$\text{rect}\left(\frac{x}{b}\right) = \begin{cases} 0 & \text{if } |x/b| > \frac{1}{2} \\ \frac{1}{2} & \text{if } |x/b| = \frac{1}{2} \\ 1 & \text{if } |x/b| < \frac{1}{2} \end{cases}$$

is a rectangle with support length b .

As shown in the top graph of Figure 18, if $|u_o| < \frac{u_s}{2}$, the original cosine can be recovered exactly. We illustrate in both Figures 18 and 19 what happens as the frequency of the original signal rises past the Nyquist frequency. Figures 18a-d show "side views" of the situation for various, increasing values of u_o from the top down. The horizontal arrows indicate which direction the δ 's will move with increasing u_o . Figure 19 shows a "top view" as u_o increases linearly from left to right. The spectrogram has been shaded. The four vertical cuts in this figure correspond to the four situations shown in Figure 18.

In Figure 18-b, the cosine's frequency has exceeded the Nyquist rate, and δ 's from neighboring spectral orders have moved into the interpolation rectangle. We show how the various δ 's correspond with the dashed lines drawn from graph to graph. The apparent effect of a rise in u_o is a bounce in frequency, which is more apparent in Figure 19. Just as the outgoing δ 's leave the interpolation rectangle, incoming δ 's enter, moving toward the frequency origin. These two incoming δ 's continue past each other, producing another bounce in apparent frequency, as shown in Figure 18-c. When these δ 's leave, they are replaced by two more, as in Figure 18-d, and the process continues on and on. This process causes the apparent bouncing in the spectrogram illustrated in Figure 19.

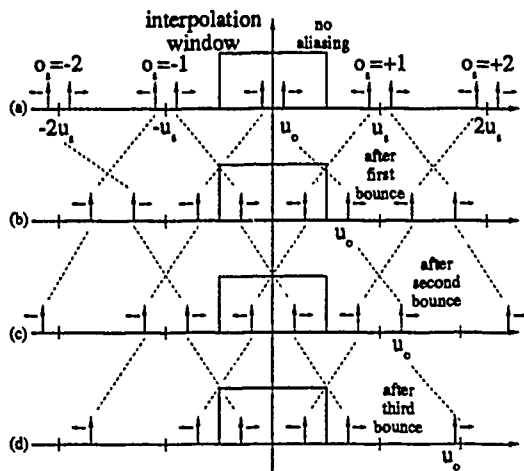


Figure 18: Aliasing causing bouncing, u_o is increasing from the top graph down

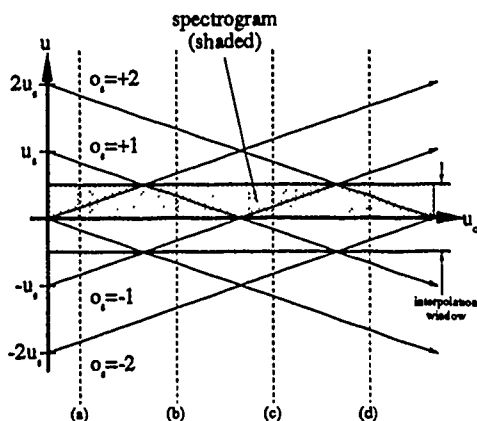


Figure 19: Aliasing causing bouncing, u_o is increasing from left to right

In Table 2 we illustrate with equations what is happening in each of the four subfigures of Figure 18. We label each situation with o_s , the spectral order which contributes the δ in the positive half of the interpolation window in frequency space. In (a), $o_s = 0$, and the cosine's frequency is below the Nyquist frequency, so the reconstruction is true to the original signal. In (b) the reconstruction is based on one δ from each of the two closest neighboring spectral orders, and $o_s = +1$. The reconstructed signal is $\cos[2\pi(u_s - u_o)x]$. Since $u_o \leq u_s$ in this case, an increase in u_o (the original signal's frequency) will cause a *decrease* in the frequency of the reconstructed signal. In (c) no new δ 's are introduced, but the two δ 's pass each other. Thus, in (c) $o_s = -1$. The reconstructed signal is $\cos[2\pi(-u_s + u_o)x]$, which is the same as case (b) (because $\cos(-t) = \cos(t)$). However, in (c) $u_o \geq u_s$, so an increase in u_o causes an *increase* in the frequency of the reconstructed signal. The transition from (c) to (d) is like the transition from (a) to (b), thus the frequency of the reconstructed signal decreases again with increasing u_o . In general, the frequency of the reconstructed cosine is given by

$$u = \begin{cases} u_o & \text{if } o_s = 0 \\ o_s u_s - \text{sgn}(o_s) u_o & \text{otherwise.} \end{cases} \quad (5)$$

where o_s is the spectral order contributing a δ to the positive half of the interpolation function, u_s is the sampling frequency, and

$$\text{sgn}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ +1 & \text{if } x > 0. \end{cases}$$

5.2 Unfolding the Spectrogram

Of course, it would be better to have no aliasing in the spectrogram. We could then get an accurate idea of the true signal at every point. We can think of the spectrogram as a distorted, windowed version of an ideal, space/frequency representation – the ideal spectrogram. The ideal spectrogram's frequency axis extends from zero to infinity, and it does not suffer from aliasing. We can see from the analysis in the previous subsection that the actual spectrogram of a simple sinusoid whose frequency is changing is a folded version of the ideal spectrogram. This is illustrated in Figure 20. The folds occur at positive, integer multiples of the Nyquist frequency, $u_s/2$. In the ideal spectrogram, the frequency peak continues to grow with the frequency of the underlying signal, while in the actual spectrogram aliasing causes the apparent frequency to bounce between zero and the Nyquist frequency.

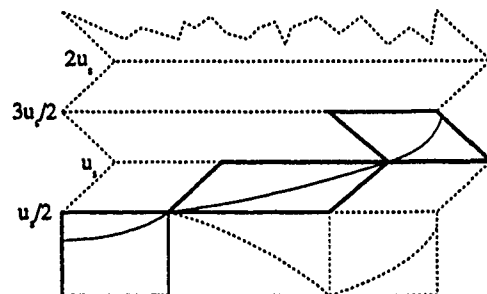


Figure 20: Folding the ideal spectrogram to show aliasing

In Figure 21 we show an unfolded version of the spectrogram in Figure 16. The unfolded spectrogram gives a true indication of the signal's frequency, even beyond the Nyquist limit. Unfolding the spectrogram of a signal with overtones, like that in Figure 17, would not be as simple. Multiple peaks in the same column may come from different folds of the ideal spectrogram. The key is to determine which fold a given peak came from. In the next section, we propose an algorithm for this based on computer-controlled zooming of the lens.

6 Lens Parameters and the Spectrogram

Much research in "active vision" concerns the control of the three lens parameters: zoom, focus, and aperture. We show in this section how these parameters affect the spectrogram, which in turn provides new insights into how they affect the image. This point of view leads to algorithms which let us deduce intrinsic scene parameters by purposefully altering the lens settings.

6.1 Zoom

6.1.1 How Zoom Affects the Spectrogram

In *equifocal* camera lenses (such as most one-touch zoom lenses) a change in zoom can be modeled as simply a change

u_o	o_s	frequency domain reconstruction	space domain reconstruction
(a) $0 \leq u_o \leq u_s/2$	0	$\frac{1}{2} [\delta(u + u_o) + \delta(u - u_o)]$	$\cos[2\pi u_o x]$
(b) $u_s/2 \leq u_o \leq u_s$	+1	$\frac{1}{2} [\delta(u + u_s - u_o) + \delta(u - u_s + u_o)]$	$\cos[2\pi(u_s - u_o)x]$
(c) $u_s \leq u_o \leq 3u_s/2$	-1	$\frac{1}{2} [\delta(u - u_s + u_o) + \delta(u + u_s - u_o)]$	same as above
(d) $3u_s/2 \leq u_o \leq 2u_s$	+2	$\frac{1}{2} [\delta(u + 2u_s - u_o) + \delta(u - 2u_s + u_o)]$	$\cos[2\pi(2u_s - u_o)x]$
\vdots	\vdots	\vdots	\vdots
$(o_s - \frac{1}{2})u_s \leq u_o \leq o_s u_s$	$o_s > 0$	$\frac{1}{2} [\delta(u + o_s u_s - u_o) + \delta(u - o_s u_s + u_o)]$	$\cos[2\pi(o_s u_s - u_o)x]$
$-o_s u_s \leq u_o \leq (-o_s + \frac{1}{2})u_s$	$o_s < 0$	$\frac{1}{2} [\delta(u - o_s u_s + u_o) + \delta(u + o_s u_s - u_o)]$	same as above

Table 2: Analytic expressions of Figure 18



Figure 21: Unfolded version of spectrogram in Figure 16

in magnification. We can imagine the situation in Figure 22-a where the section of the signal which falls on the center window of the spectrogram extends from $-\frac{l}{2}$ to $\frac{l}{2}$. We will arbitrarily call the magnification here one, and we will say that the entire portion of the signal seen by the camera is of length L . Both l and L are measured on the image plane. If there are n pixels in the spectrogram window, the sampling frequency is $\frac{n-1}{l}$ pixels per unit distance, making the Nyquist frequency $\frac{n-1}{2l}$. Since the spectrogram extends in frequency from zero to the Nyquist frequency, the spectrogram resulting from this signal will cover the region indicated by the short, wide box in Figure 23.

If the magnification M is changed, a larger or smaller portion of the original signal will be contained by each window. In Figure 22-b we have indicated the effect of an increase in magnification, showing how a smaller part of the signal is now imaged. The section of the signal which falls on the central window now extends from $-\frac{l}{2M}$ to $\frac{l}{2M}$, and the entire signal seen by the camera covers $\frac{L}{2M}$ to $\frac{L}{2M}$. The magnified window is spread out over the same number of pixels as before, so the Nyquist frequency is now $\frac{M(n-1)}{2l}$ pixels per unit distance.

The spectrogram after the magnification change is shown in Figure 23. For an increase in magnification, the spectrogram covers more in frequency but less in space. The "area" of the spectrogram (actually a unitless quantity, "spatial dynamic range") is $\frac{L(n-1)}{2l}$ and is independent of the magnification. Thus for changes in zoom, there is a direct tradeoff

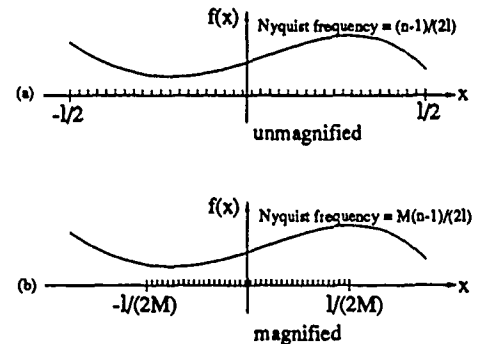


Figure 22: Effect of zooming on imaged signal

between coverage in space and spatial frequency. These arguments also apply to the four-dimensional hypervolume of the spectrogram of a two-dimensional signal.

6.1.2 Dealiasing With Zoom Changes

A slight change in zoom can be used to find the true, unaliased frequency of a sinusoid, because aliased frequencies from different spectral orders respond differently to changes in magnification. Since image textures can be decomposed into simple sinusoids, we could use two images taken at slightly different zoom settings to dealias texture images.

Suppose as above that we have a 1-D image of a cosine of frequency u_o cycles/pixel sampled at a rate of u_s cycles/pixel. The cosine may be sampled above or below the Nyquist rate. Referring to Figure 19, we can see there will be only one spectral order contributing a δ to the spectrogram (because the spectrogram only shows positive frequencies up to $u_s/2$). The apparent frequency of the unmagnified ($M = 1$) signal, u_1 , is given by Equation 5, i.e.

$$u_1 = \begin{cases} u_o & \text{if } o_s = 0 \\ o_s u_s - \text{sgn}(o_s) u_o & \text{otherwise.} \end{cases} \quad (6)$$

If the lens is zoomed slightly such that the magnification is changed to M , the sampling frequency (measured in cycles/pixel of the *unmagnified* image) will be $M u_s$ cycles/pixel, where u_s is the sampling frequency on the unmagnified image. The apparent frequency of the cosine will then be

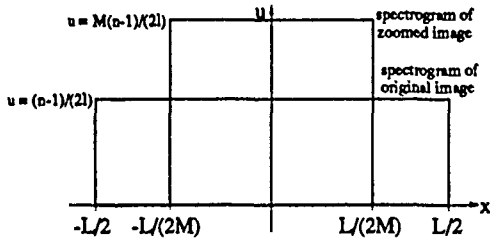


Figure 23: Effect of zooming on spectrogram dimensions

$$u_2 = \begin{cases} u_o & \text{if } o_s = 0 \\ o_s M u_s - \text{sgn}(o_s) u_o & \text{otherwise.} \end{cases} \quad (7)$$

We can eliminate u_o from Equations 6 and 7 by subtracting. Solving this difference for o_s gives

$$o_s = \frac{u_2 - u_1}{u_s(M - 1)}.$$

We note that this equation applies for both $o_s = 0$ and $o_s \neq 0$. Thus, the difference in apparent frequency between the two images is proportional to the spectral order o_s . After solving for o_s , we can use Equation 6 or 7 to solve for u_o , which is the true frequency of the signal. The dealiasing does not require the solution of a correspondence problem, since the two signals are related by a simple difference in magnification.

An implicit assumption here is that o_s remains the same in both images. This will be true for small changes in magnification unless the δ is very close to either extreme of the interpolation window and the zoom change causes it to be replaced by another δ .

We have applied this technique to the image of the receding plate in Figure 16. We show a split version of the image in Figure 24. On the top is the unmagnified image, and on the bottom is the same image magnified by $M = 1.075$. It is easily seen how the moire patterns shift. Figure 25 shows the "subpixel" frequency peaks from the spectrograms of the center rows of the two images. The frequency data from the magnified image has been adjusted so it is shown in terms of the space and frequency units of the unmagnified image. The dotted line shows the dealiasied frequency based on the technique outlined above. Except for the glitches at the frequency extremes, the figure shows correctly the dealiasied frequency. Thus, the spectrogram has been dealiasied without detailed *a priori* knowledge of the scene.

6.2 Focus and Aperture

Changes in the lens' focus and aperture combine to change the point spread function (psf) of the lens, which can be easily visualized with the spectrogram. (The psf is a function which can be convoluted with an ideal, sharp signal to model the effects of blur.) In general, points in sharper focus will show more high frequencies than if they are blurred. A smaller aperture tends to have the same general effect as sharper focus.

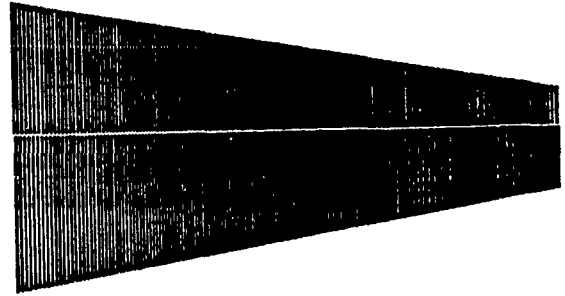


Figure 24: Horizontally split image of aliased plate and magnified aliased plate

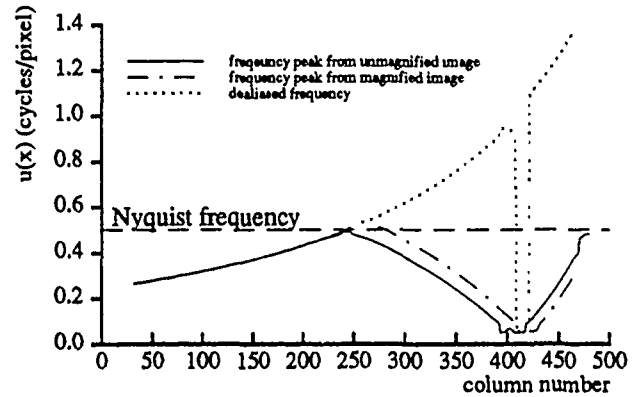


Figure 25: Dealiasing with magnification change

In fact, in the pinhole model we have been using (Figure 12), the aperture is infinitesimally small, meaning that every point in the scene is in perfect, sharp focus.

We will generalize the pinhole model by introducing a single, thin lens with a variable aperture as shown in Figure 26. The aperture of the lens is a , the focal length of the lens is b , and the distance to the image plane remains d . We can approximate the effects of focus and aperture with geometric optics. Each point in the scene with a different value of z_{3D} will be in sharp focus at only one point behind the lens. This point, z , is given by the Gaussian Lens Law: $\frac{1}{z} + \frac{1}{-z_{3D}} = \frac{1}{b}$. If the image plane is not at the proper distance behind the lens, i.e. $d \neq z$, the point will be spread into a blur circle. Using geometric optics, the radius of the blur circle is given by

$$r(z_{3D}) = \frac{ad}{2} \left| \frac{1}{b} - \frac{1}{d} + \frac{1}{z_{3D}} \right|.$$

A point can be out of focus by having the image plane in front of or behind the point of best focus. The equation above applies to both cases. $r(z_{3D})$ goes to zero when $\frac{1}{d} + \frac{1}{-z_{3D}} = \frac{1}{b}$, which is a restatement of the Gaussian Lens Law above. In the one-dimensional imaging case illustrated here, the shape of the blur "circle" is actually a rectangle of width $2r(z_{3D})$. Thus, the point spread function of the 1D camera system is

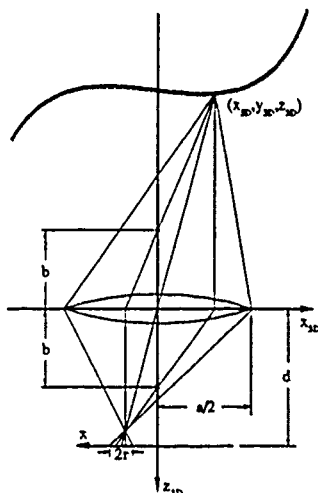


Figure 26: Geometry of 1D image formation through thin lens

$$h(x, z_{3D}) = \frac{1}{2r(z_{3D})} \text{rect} \left[\frac{x}{2r(z_{3D})} \right],$$

where we have normalized so the area under the `rect()` is one.³ The corresponding transfer function, H , is the Fourier transform of h :

$$H(u, z_{3D}) = \text{sinc} [2ur(z_{3D})] .$$

In order to calculate the effect of $h(x, z_{3D})$ on the spectrogram, we suppose there exists a function $f(x)$ which is an unblurred, pinhole projection of the scene. The new image, $f_h(x)$, taking into account the point spread function, is a convolution of the unblurred image with h . Thus,

$$f_h(x) = \int_{-\infty}^{\infty} f(\zeta) h(x - \zeta, z_{3D}) d\zeta,$$

where the z_{3D} is the one corresponding to ζ on the image plane. This equation holds for changes in the camera's aperture. It does not apply for change in the focus distance d , because this causes a change in magnification as well as a change in the point spread function.

The point spread function h is *not* space-invariant, because it depends on the depth of the surface. This means that its effect cannot be described accurately by multiplication in the frequency domain. If h were space-invariant, e.g. due to integrating over the surface of the pixels, then the effect on the spectrogram would be simple to describe: each windowed Fourier transform would be multiplied by the Fourier transform of the point spread function. This is also approximately true for the space-variant point spread function if the surface depth varies slowly and/or the window used for the spectrogram is small. Then we have

$$S_{f,h}(x,u) \approx \left| \left[e^{-j2\pi xu} \tilde{W}_l(u) \right] * F(u) * H(u, \bar{z}_{3D}) \right|^2,$$

³This psf ignores three optical effects. One is diffraction, whose magnitude is much smaller than defocus effects in typical TV images. The second is the fact that points which are occluded in the pinhole image can actually be seen by parts of the lens in an image with a finite aperture. The third is that, by normalizing the area of the psf to one, we are ignoring the most obvious effect of a change in aperture, a change in the overall brightness of the image.

where \bar{z}_{3D} is a representative depth value for the region centered at x , and $F(u)$ is the Fourier transform of the unblurred image. Each windowed Fourier transform has associated with it its own transfer function which depends on the approximate depth of the region within the window.

This is the approximation used for most depth from focus and depth from defocus algorithms in computer vision. Following Krotkov's [Krotkov, 1987] depth from focus algorithm, the spectrogram can be used as a criterion function to calculate the point of best focus over several images taken at different focus settings. The setting closest to perfect focus is the one which gives the most high frequency energy in the spectrogram at that point. Knowing this setting along with a precalibrated table of focus distances, the depth to all points in the scene can be calculated. Pentland [Pentland, 1985] uses a spectrogram, essentially, to calculate depth from defocus based on only two focus settings. He uses the two spectrograms to calculate directly the depth to each scene point by calculating the width of the psp.

Formulating the effects of the psf in terms of the spectrogram is a natural way to reason about the space-variant nature of the transfer function. For example, it reveals how precisely each point can be focused. Points in the scene with no high frequencies will never show high frequencies no matter how well they are focused, meaning that a focusing criterion function based on frequency would not be sensitive to such points. Another issue is the separation of the space-invariant part of the psf (due to, say, pixel averaging and the camera electronics) from the space variant part. It may be that the space-invariant psf is so large that depth effects are insignificant.

7 Other Issues

7.1 Variable Window Size

A constant window size for the spectrogram means that the Fourier transforms cover a different number of wavelengths of each constituent frequency. That is, a window size l over a signal of frequency u covers lu wavelengths or periods of the signal. In detecting repetitions at different frequencies, it makes intuitive sense that the detector window should cover a predetermined number of wavelengths rather than a predetermined length or area. This intuition is based on the feeling that a texture pattern is one comprised of some minimum number of similar elements rather than some minimum sized region. The conventionally defined spectrogram uses a constant window size, which means that for higher frequency signals, more wavelengths of the signal will be included in the window than for lower frequency signals. Thus the localization (spatial resolution) of the *constant-window spectrogram* is effectively reduced at higher frequencies, because the window is spread out over more wavelengths.

We propose adding another dimension to the spectrogram which indicates the window size l . We define the *3D spectrogram* given by

$$S_f(x, u, l) = \left| \int_{-\infty}^{\infty} w(\alpha - x, l) f(\alpha) e^{-j2\pi u \alpha} d\alpha \right|^2,$$

which covers all possible (positive) window sizes.

The 3D spectrogram is a great deal of data which is highly redundant. The constant-window spectrogram, $S_f(x, u)$, is a slice of $S_f(x, u, l)$ with $l = \text{constant}$. The problem with a constant l is that, as we mentioned above, the number of wavelengths included in the window varies with frequency. A more reasonable slice through the 3D spectrogram is to have $l \propto 1/u$, which means that the window width will shrink with

decreasing wavelength. This tends to make the spectrogram scale-invariant, in that the detector window will cover a constant number of elements of a given wavelength independent of their spacing frequency. We call this the *variable-window spectrogram*.

We show an example of the the variable-window spectrogram in the bottom half of Figure 27, which can be compared to the traditional, constant-window spectrogram in the top half of the same figure. The variable-window spectrogram has window size $l = 10/u$. One notable aspect of the variable-window spectrogram is the large spreading of the higher frequencies. This is due to the familiar effect in Fourier analysis of a smaller spatial domain window giving more spread in the frequency domain. Thus, the variable-window spectrogram provides greater spatial resolution at a cost of frequency resolution. The spreading of the high frequencies leads us to a conjecture that a nonlinear sampling in frequency may be appropriate for the variable-window spectrogram. In the case of $l \propto 1/u$, the frequency sampling interval should get larger as the frequency increases.

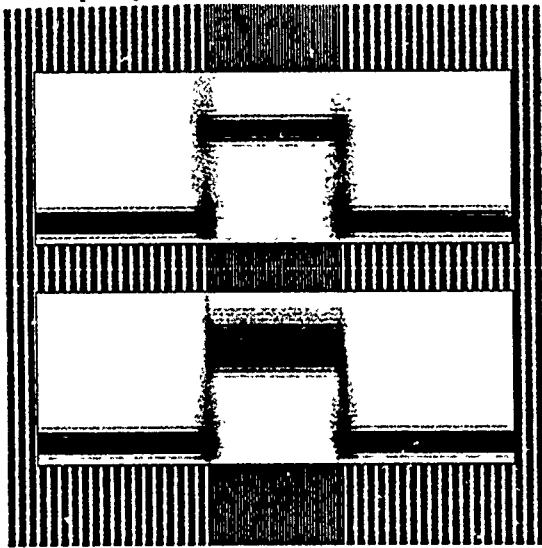


Figure 27: Variable window (bottom) vs. constant window (top) spectrogram

7.2 Repetition and Image Matching

Image matching is important for 3D stereo and motion sequence analysis. In these tasks, matches are found by shifting one image to match the other; the amount of shift needed at each point reveals the 3D structure of the scene. If a portion of the image is uniform with no features, then matching is impossible; if features are present, a match can be obtained. In the ideal case of a step intensity edge, a match can be made with infinite precision. Usually, heuristic measures of potential precision are used, such as finding "feature points". But here, as in other spatial vision tasks, the spectrogram is useful to quantify this effect. The match precision available at any point in the image is limited by the highest spatial frequency present at that point. This is illustrated in Figure 28: a narrow bump or step edge can be matched with greater precision than the shallow, broad bump in the signal. This is reflected in the higher spatial frequency content for the more precise features, as shown in Figure 29. The figure shows an image whose scanlines are all identical to the intensity profile shown in Figure 28. On top is the variable-window spectrogram of one scanline, which shows that the step edge and narrow bump have higher spatial frequencies than the broad

bump, and would therefore give higher precision matches. This spectrogram has window size $l = 5/u$.

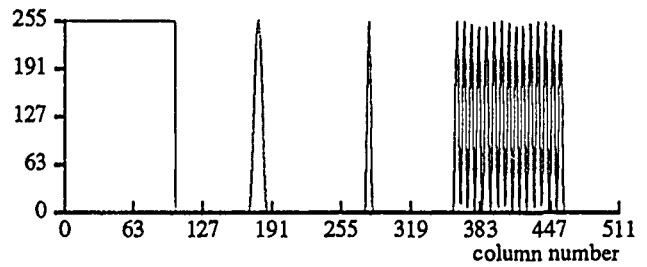


Figure 28: Intensity profile to be matched

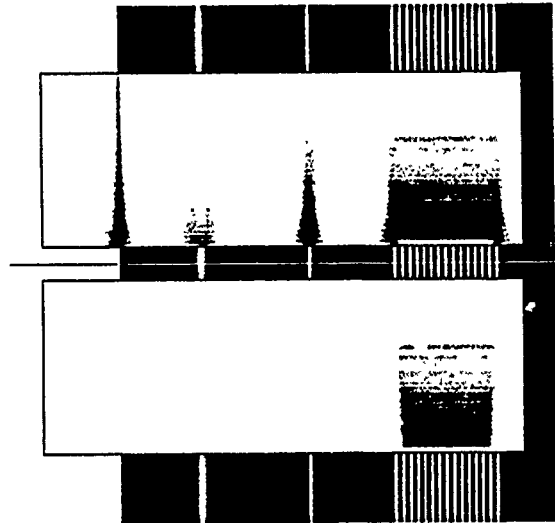


Figure 29: Spectrogram and repeatogram for image matching

The spectrogram also provides insight about another aspect of image matching: False matches. One of the hardest problems in motion or stereo vision is to know whether a potential feature match is a real, dependable correspondence, or whether it is a false match with a different feature in the other image. For example, in Figure 28 above, the right side shows several bumps closely spaced – if there were a large uncertainty in the displacement between images, the wrong bumps might be matched with each other.

There is a clear relationship between false match potential and frequency content, for a false match must be characterized by a repetition in the image signal at the corresponding scale. Yet, each bump in the group on the right of the figure has the same profile as the isolated bump just to their left. So, the distinction must be more complex than just examination of the spectrogram at each point. The key is that false match potential implies not just high frequency content, but a real repetition of the image data, which means frequency content that *persists* over more than one wavelength of the underlying sinusoid. Thus, to detect false matches (or image structure repetition in general), one must search the spectrogram for frequency content that persists over long intervals in the spatial dimension.

To represent this, we propose a new transform we call the *repeatogram*, which is derived from the spectrogram as follows: At each point x and frequency u , the repeatogram $R(x, u)$ is the minimum magnitude of the spectrogram over an interval centered at x and extending for k wavelengths of the underlying sinusoid on either side of x , i.e.

$$R(x, u) = \min [S(x', u)] \text{ for } x' \in [x - (k/2u), x + (k/2u)].$$

We call this the k -repeatogram, and note that for $k \geq 1.5$, there must be at least two relative maxima or relative minima of the underlying sinusoid within the interval of examination; for $k \geq 2$, there must be at least two of each. In general, where $R(x, u)$ is high, a real repetitive structure exists in the image, with period $1/u$ pixels wide.

For a spectrogram with a nonzero window size, these considerations must be modified slightly, because a window can contain part of a repetition before it is actually centered on the repetition. Specifically, for a window of length l and a repetition over the range $[x_1, x_2]$, the spectrogram will show a reaction to the repetition over the range $[x_1 - l/2, x_2 + l/2]$. The matter is further complicated by the fact that most windows, including our default window in Equation 2, drop off toward zero at their ends, meaning that the spectrogram will be fairly insensitive to the repetition until the window is almost centered over the repetition. The effect of these complicating factors is that the choice of k for the k -repeatogram is dependent on the window size and shape.

Figure 29 shows, in the bottom half, the 4-repeatogram for the profile in Figure 28. As seen in this figure, the repeatogram makes it quite clear that the features on the right of the signal exhibit real repetition, while the isolated bump of the same shape does not.

With the repeatogram and the spectrogram, we therefore have a powerful pair of tools for predicting the accuracy and precision of matching displaced images. At any point, the highest significant frequency content in the spectrogram tells how precise a match can possibly be obtained; the highest frequency with significant content in the repeatogram tells the maximum displacement search window size that can be tolerated before there is a danger of obtaining a false match.

8 Conclusion

For now, we are continuing to develop useful theories for computer vision based on space/frequency representations rather than to bring any one of the techniques described above to completion. Our work thus far has shown the versatile power of the representation rather than demonstrated any end-to-end analysis. At this point, we believe the most effort is needed in assessing the potential for this kind of approach to vision rather than in trying to build specific programs to analyze this or that particular phenomena. We have presented a few experimental results, but they are meant to be suggestive rather than definitive algorithms. Instead, we wish to point out the breadth of this approach to low-level spatial vision, and in particular its potential contribution for:

General Vision As an alternative to traditional edge-finding and region-grouping methods, which are known to be very brittle and noisy. The spectrogram also captures the 3D shape and 2D texture characteristics of surfaces.

Matching Problems As a way of showing specifically what displacement of stereo or motion can be tolerated for reliable matching at each point in the image.

Active Camera/Lens Control As a way of formulating the constraints and goals for purposeful zoom, focus, and camera motion.

This line of investigation, obviously, is far from complete. In particular, we see challenges in the analysis of complex textures such as the Brodatz patterns rather than simple sinusoid and square waves; expressing the relationship between 3D

surface texture, radiometry (lighting and reflection), and 2D image texture; and the development of effective algorithms to compute and analyze the spectrogram. It may also turn out that the spectrogram is primarily useful not as a representation to use in the vision system itself, but rather as a way of understanding the theory behind an implementation that uses, for example, a small set of Gabor functions instead. In any event, we believe that the power of the space/frequency distribution will make it possible to develop far more comprehensive methods for low-level spatial vision than the current, limited, techniques allow.

9 Acknowledgements

Thank you to Sheryl Young for suggesting the method for detecting aliasing by changing zoom as described in Section 6.1.2.

References

- [Bajcsy and Lieberman, 1976] Ruzena Bajcsy and Lawrence Lieberman. Texture gradient as a depth cue. *Computer Graphics and Image Processing*, 5:52-67, 1976.
- [Boashash, 1988] Boualem Boashash. Note on the use of the wigner distribution for time-frequency signal analysis. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(9):1518-1521, September 1988.
- [Bovik et al., 1990] Alan Conrad Bovik, Marianna Clark, and Wilson S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):55-73, January 1990.
- [Brodatz, 1966] Phil Brodatz. *Textures: A Photographic Album for Artists and Designers*. Dover Publications, 1966.
- [Choi and Williams, 1989] Hyung-Il Choi and William J. Williams. Improved time-frequency representation of multicomponent signals using exponential kernels. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(6):862-871, June 1989.
- [Claasen and Mecklenbrauker, 1980a] T.A.C.M. Claasen and W.F.G. Mecklenbrauker. The wigner distribution - a tool for time-frequency signal analysis, part i: Continuous-time signals. *Phillips Journal of Research*, 35(3):217-250, 1980.
- [Claasen and Mecklenbrauker, 1980b] T.A.C.M. Claasen and W.F.G. Mecklenbrauker. The wigner distribution - a tool for time-frequency signal analysis, part ii: Discrete-time signals. *Phillips Journal of Research*, 35(4/5):276-300, 1980.
- [Claasen and Mecklenbrauker, 1980c] T.A.C.M. Claasen and W.F.G. Mecklenbrauker. The wigner distribution - a tool for time-frequency signal analysis, part iii: Relations with other time-frequency signal transformations. *Phillips Journal of Research*, 35(6):372-389, 1980.
- [Connors and Harlow, 1980] Richard W. Connors and Charles A. Harlow. A theoretical comparison of texture algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2(3):204-222, May 1980.
- [Daugman, 1985] John G. Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America A*, 2(7):1160-1169, July 1985.

- [Daugman, 1988] John G. Daugman. Complete discrete 2-d gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(7):1169-1179, July 1988.
- [Dyer and Rosenfeld, 1976] Charles A. Dyer and Azriel Rosenfeld. Fourier texture features: Suppression of aperture effects. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(10):703-705, October 1976.
- [Fogel and Sagi, 1989] I. Fogel and D. Sagi. Gabor filters as texture discriminator. *Biological Cybernetics*, 61(2):103-113, June 1989.
- [Gabor, 1946] D. Gabor. Theory of communication. *The Journal of the Institution of Electrical Engineers, Part III*, 93(21):429-457, January 1946.
- [Gramenopoulos, 1973] Nicholas Gramenopoulos. Terrain type recognition using earth-resources images. In *Symposium on Significant Results Obtained from the Earth Resources Technology Satellite*, pages 1229-1241. NASA Scientific and Technical Information Office, March 1973.
- [Haralick, 1979] Robert M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786-804, May 1979.
- [Harris, 1978] Fredric J. Harris. On the use of windows for harmonic analysis with the discrete fourier transform. *Proceedings of the IEEE*, 66(1):51-83, January 1978.
- [Heeger, 1988] David J. Heeger. Optical flow using spatiotemporal filters. *International Journal of Computer Vision*, 1(4):279-302, January 1988.
- [Horn, 1968] Berthold Horn. Focusing. Artificial Intelligence Memo 160, MIT, May 1968.
- [Jacobson and Wechsler, 1988] Lowell D. Jacobson and Harry Wechsler. Joint spatial/spatial-frequency representation. *Signal Processing*, 14(1):37-68, January 1988.
- [Jau and Chin, 1988] Y.C. Jau and Roland T. Chin. Shape from texture using the wigner distribution. In *Computer Vision and Pattern Recognition*, pages 515-523. Computer Society Press, June 1988.
- [Kender, 1980] John R. Kender. Shape from texture. Technical Report CMU-CS-81-102, CMU, November 1980.
- [Kirvida, 1976] Leonard Kirvida. Texture measurement for the automatic classification of imagery. *IEEE Transactions on Electromagnetic Compatibility*, EMC-18(1):38-41, February 1976.
- [Krotkov, 1987] Eric Krotkov. Focusing. *International Journal of Computer Vision*, 1(3):223-237, May 1987.
- [Mallat, 1989] Stephane G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674-693, July 1989.
- [Marcelja, 1980] S. Marcelja. Mathematical description of the responses of simple cortical cells. *Journal of the Optical Society of America*, 70(11):1297-1300, November 1980.
- [Matsuyama et al., 1983] Takashi Matsuyama, Shu-Ichi Miura, and Makoto Nagao. Structural analysis of natural textures by fourier transformation. *CVGIP*, 24:347-362, 1983.
- [Matthies, 1989] Larry Matthies. Dynamic stereo vision. Technical Report CMU-CS-89-195, Carnegie Mellon University, October 1989.
- [Pentland, 1985] Alex P. Pentland. A new sense for depth of field. In Aravind Joshi, editor, *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 988-994. Morgan Kaufmann Publishers, Inc., August 1985.
- [Pentland, 1988] Alex Pentland. The transform method for shape from shading. Media Lab Vision Sciences Technical Report 106, MIT, July 1988.
- [Reed and Wechsler, 1990] Todd R. Reed and Harry Wechsler. Segmentation of textured images and gestalt organization using spatial/spatial frequency representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):1-12, January 1990.
- [Shafer et al., 1990] Steven A. Shafer, Takeo Kanade, Gunder J. Klinker, and Carol L. Novak. Physics-based models for early vision by machine. In *SPIE Conference on Perceiving, Measuring, and Using Color*, #1250. SPIE, February 1990.
- [Stevens, 1981] Kent A. Stevens. The information content of texture gradients. *Biological Cybernetics*, 42(2):95-105, November 1981.
- [Tanimoto and Pavlidis, 1975] S. Tanimoto and T. Pavlidis. A hierarchical data structure for picture processing. *Computer Graphics and Image Processing*, 4(2):104-119, June 1975.
- [Turner, 1986] M.R. Turner. Texture discrimination by gabor functions. *Biological Cybernetics*, 55(1):71-82, October 1986.
- [Van Gool et al., 1985] L. Van Gool, P. Dewaele, and A. Oosterlinck. Texture analysis anno 1983. *Computer Vision, Graphics, and Image Processing*, 29:336-357, 1985.
- [Wechsler, 1980] Harry Wechsler. Texture analysis - a survey. *Signal Processing*, 2:271-282, 1980.
- [Weszka et al., 1976] Joan S. Weszka, Charles R. Dyer, and Azriel Rosenfeld. A comparative study of texture measures for terrain classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(4):269-285, April 1976.
- [Witkin, 1981] Andrew P. Witkin. Recovering surface shape and orientation from texture. *Artificial Intelligence*, 17:17-45, June 1981.
- [Zhao et al., 1990] Y. Zhao, L. Atlas, and R. Marks. The use of cone-shaped kernels for generalized time-frequency representations of nonstationary signals. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, (to appear) June 1990.

Least Median of Squares Based Robust Analysis of Image Structure

Peter Meer, Doron Mintz, and Azriel Rosenfeld
Computer Vision Laboratory, Center for Automation Research,
University of Maryland, College Park, MD 20742-3411

ABSTRACT

Representation of an image by piecewise polynomial surfaces is an important computer vision task. Unfortunately the traditional least squares based techniques are prone to error when applied to nonhomogeneous regions. We introduce a new, robust algorithm which recovers the fit corresponding to the absolute majority (at least 51 percent) of the pixels in the processing window. The algorithm uses the least median of squares (LMedS) estimator recently introduced in the statistics literature. We show that the estimator may yield incorrect results when applied to images, and propose a two-stage procedure for the local description of the image structure and of the corrupting noise. Robust region growing can also be performed with the same technique. The goal of this paper is to introduce the LMedS paradigm to the computer vision community, the possible applications are not restricted to the ones treated here.

1 Introduction

The information in an image is carried by two types of primitives: homogeneous patches and discontinuities. *Homogeneous patches* are regions in which the values can be characterized by a small number of parameters. In this paper homogeneity is defined relative to a polynomial surface fit, i.e., the parameters are the coefficients of a polynomial taking values on a grid with sites corresponding to integers. *Discontinuities* occur where homogeneous patches meet. The two primitive types are dual to each other. Homogeneous patches are delineated by discontinuities, while discontinuities are defined by pairs of adjacent homogeneous patches. When primitives of one type are known those of the other type are immediate.

Human visual perception offers several examples of the interaction between discontinuities and homogeneous patches. In the Craik-O'Brien illusion (Cornsweet, 1970), if a local discontinuity similar to the smoothed first derivative of a step is introduced in the middle of a uniform field, a difference in brightness between the two entire halves of the field is perceived. The visual system couples shadows (homogeneous patches) to their generating objects by analyzing only the borders

(discontinuities) (Cavanagh and Leclerc, 1989).

In the most often employed image analysis method the relationships among delineated homogeneous patches are examined. (Note that we consider here only the lowest-level image analysis in which the assumed model is that of a local polynomial surface structure.) The duality of the two primitive types allows two different approaches. In the discontinuity-based approach first the discontinuities are detected by local operations and the regions bounded by contiguous discontinuities are then defined as homogeneous patches. A similar principle can be used to achieve high image compression ratios (Kunt *et al.*, 1987). In the dual, homogeneity-based approach, first the homogeneous patches are found and the discontinuities are obtained where they meet. In Section 2 we review recent results obtained by each approach and argue that the homogeneity-based approach is more appropriate for computer vision problems.

Detection of homogeneous patches first requires, however, powerful algorithms. When seeking regions with the same polynomial fit, the underlying assumption is that only one such surface is present in the processing window. Thus when the analyzed region contains discontinuities, due to the incorrect model erroneous decisions may be made. To reduce the dependence of the performance on the initial assumptions robust detection techniques must be employed. In Section 3 we discuss the necessity for robust procedures and their required characteristics, and introduce least median of squares estimators.

The specific nature of the image data requires adaptation of the least median of squares estimators. In Section 4 we describe a robust, homogeneity-based image analysis algorithm making use of the least median of squares estimators. The algorithm recovers the parameters of the polynomial fit and supplies the description of the local image structure as well as of the corrupting noise. Recovery of extended homogeneous regions is achieved by a fusion/region-growing procedure using the same least median of squares based technique.

The goal of this paper is to introduce the paradigm of the least median of squares robust estimators in computer vision. We have chosen the analysis of noisy images modeled as piecewise-polynomial surfaces to illustrate the potential of the technique. Several other computer vision applications exist in which the

properties of least median of squares estimators are beneficial. We discuss some of them in Section 5.

2 The Two Approaches to Image Analysis

In this section we review the recent literature on image structure analysis. Section 2.1 treats discontinuity-based approaches, and Section 2.2 homogeneity-based approaches.

The feature property of interest for us is the local polynomial fit to the data. That is, can a small region in the image be represented as a polynomial surface? Koenderink and van Doorn (1982) have shown that elliptic and hyperbolic patches suffice to decompose an object in the physical world. In computer vision, partly due to quantization effects, higher order fits are also employed. Haralick *et al.* (1983) used bicubic surfaces to build the topographical primal sketch of intensity data, Besl and Jain (1988) used biquartic polynomials for the description of range images.

2.1 Discontinuity-based Approaches

Discontinuities in an image can result from adjacency of any two polynomial surfaces. A step edge appears when the two polynomials are both of degree zero, roof edges are the result of two first order surfaces with common boundary values, etc. The number of possible discontinuity profiles is very large.

Optimality of several recently proposed edge detectors (Canny, 1986; Bergholm, 1987) is contingent upon the presence of the discontinuity profile they were designed for, usually a step edge. These detectors incorporate differentiation of the image, an operation whose efficiency decreases steeply with the amount of noise present. To improve performance in noisy environments smoothed derivatives are employed, trading off the accuracy of discontinuity localization. The optimal amount of smoothing is difficult to assess and multiresolution approaches are employed in which results from several scales are combined.

By allowing several discontinuity profiles to be present in the image, profile dependent edge detection methods become unwieldy. The difficulties are only increased if non-stationary noise is present in the image. In this case smoothing should be position dependent, resulting in variable degradation of the discontinuities in the original image. Blurring in the image formation process also contributes to the spread and diversification of the discontinuity profiles.

The limitations of edge detection methods based on differentiation are well understood and different approaches are often sought. Lee *et al.* (1988) and Pavlidis and Lee (1989) proposed a residual based procedure. The residual is defined as the difference between the value of the pixel and its estimate obtained by smoothing in a small neighborhood. Smoothing can be achieved either by convolution or by regularization. The discontinuities are localized at the zero-crossings of the residual image. Lee *et al.* (1988) established a necessary and sufficient condition

to have a zero-crossing correspond to the correct location of a discontinuity in the original image. Without additional processing, however, the detectable discontinuities are restricted to steps in the image, i.e., the two functions around the discontinuity cannot have a common boundary value. Analysis of the necessary condition for the presence of a zero-crossing (Theorem 2.1 in Lee *et al.*) shows a strong dependence of the results on the shape of the image around the discontinuity. Recently Chen *et al.* (1989) extended the method by taking into account the shape of the autocorrelation function around discontinuities.

In a more general framework, Lee (1988, 1989) has proved that discontinuities in one dimension can be optimally detected in noisy signals by using derivatives of splines as matched filters. For two-dimensional images the procedure has to be applied independently in the row and column directions.

Analysis of the residuals is also employed in change detection algorithms (see the book of Basseville and Benveniste, 1986, for a collection of theoretical papers on the subject). The underlying autoregressive model is a composite one; it is assumed that the description of the data may have changed from one model to another at an unknown position in the processing window. The location of the transition is determined by maximizing the likelihood ratio between the hypotheses "transition present" and "no transition present". To discriminate a transition, however, the autoregressive model on one side of the discontinuity should be known a priori. This constraint implies sequential processing along one dimension at a time.

Basseville *et al.* (1981) applied the change detection method in a line-by-line horizontal scan to discriminate step edges in images. The potential edges were then smoothed by a Kalman filtering based edge-following scheme along the vertical direction. A similar technique was proposed by Dattatreya and Kanal (1988). Chakraborti and Misra (1987) compared several transition detection algorithms employing autoregressive modeling of the local image structure. Note that in this paper we do not discuss the problem of texture segmentation where autoregressive models are also frequently employed.

A simple CUSUM (cumulative sum) test can be derived from the likelihood ratio for detection of jumps in the mean if the perturbing noise is zero mean Gaussian (see Chapter 1 in Basseville and Benveniste, 1986). These sums involve only the values of the samples, the a priori known mean before the transition, and the smallest significant jump size.

Chow's (1960) test comparing the parameters of two polynomial fits was employed by Leclerc and Zucker (1987) to discriminate discontinuities in images. This test too can only be applied in one dimension, separately along the rows and along the columns of an image. Polynomials are fit to the left and right neighborhoods of any pixel in the one-dimensional signals. The residuals of the fits in the neighborhoods, as well as of the fit in the combined region, are used to build an F-type statistic. The

existence of a discontinuity at that pixel is then tested by comparing the value of the statistic with a threshold corresponding to the degrees of freedom. The method was implemented multiscale and bottom-up. Some of the discontinuities found at the beginning are later eliminated by retesting their locations with increasing neighborhood size. The use of the F-test requires that the noise be of a Gaussian nature. Bottom-up discontinuity discrimination is usually less efficient at small signal-to-noise ratios where the probability of false alarms increases steeply and numerous artifacts are introduced.

We conclude this survey of "non-traditional" discontinuity detection methods by mentioning an algorithm proposed by Nalwa and Binford (1986). The existence of a transition in a neighborhood is first tested with successive fits in one and two dimensions. Potential discontinuities are then fitted with a hyperbolic tangent function to achieve subpixel accuracy localization. The image before and after the transition is taken to be constant (a zero order polynomial).

2.2 Homogeneity-based Approaches

In homogeneity-based approaches the discrimination of homogeneous surface patches is usually assisted by hypotheses about the presence of discontinuities. Therefore the distinction between these methods and discontinuity-based methods is sometimes not entirely obvious. Additional support for homogeneity-based segmentation is gained when constraints about the physical world are also taken into account. In this paper we are interested only in the low-level component of the segmentation process, and thus this part of the literature will not be surveyed.

Homogeneity is defined for our purposes as a satisfactory polynomial surface fit within a local region. A minimum size (say 3×3 pixels) must be used for the regions in order for this feature property to become meaningful. Real gray-level images, however, even in the noiseless case, do not have perfect piecewise polynomial structure. To identify polynomial surface patches the homogeneity-based methods must assume that the data arise from a piecewise polynomial image. We will not deal here with traditional region growing and "split and merge" techniques (see Haralick and Shapiro, 1985, for a review), or with methods which combine edge detection and region growing (e.g. Bajcsy *et al.*, 1986; Pavlidis and Liow, 1988).

The adaptive surface labeling method proposed by Mowforth *et al.* (1987) is an example of a simple homogeneity-based approach. The regions best fitted by a simple polynomial model are searched with decreasing window sizes and increasing model orders. The method requires a priori knowledge of the noise variance, which is used as a threshold for decisions.

Besl and Jain (1988) employed, for segmentation of range images, eight fundamental surface types defined in differential geometry by combinations of the signs of the mean and Gaussian curvatures. Each delineated surface in the image was then reduced to a seed region by applying a morphological (erosion) operation. The

seed regions were grown by iterative polynomial surface fitting with increasing model order. Yokoya and Levine (1989) proposed a similar hybrid method in which surface identification is integrated with edge detection.

Several homogeneity-based methods iteratively collapse the input image into a piecewise polynomial representation. The motivation is that once the image structure is similar to the ideal, piecewise polynomial structure, detection of discontinuities becomes more reliable. In these methods, based on the current local image structure, the values of the pixels are modified in parallel. The technique employed for the facet model (Haralick and Watson, 1981) is one of the best known. The same order polynomial fit is computed within every window of a given size to which a pixel belongs. For example, if the window size is 3×3 there are nine different fits. The updated value of the pixel is taken from the window having the smallest variance (residual power). The procedure is shown to converge, but the final result is not necessarily the best piecewise polynomial approximation of the input.

In the one-dimensional smoothing technique proposed by Heinonen and Neuvo (1988) a different rule is employed to combine the descriptions of a pixel from several windows. The value of the pixel is predicted based on different model orders in neighborhood on its left and its right. For example, the predictions can be computed assuming zero order (constant) and first order (ramp) structure in the neighborhoods. The predictions can be obtained by simple convolutions. The median of the list containing the values of the predictions and the current pixel is used as the new value. Application of the method in two dimensions, other than one dimension at a time, is not straightforward.

Excessive smoothing across discontinuities, while collapsing the input image into a piecewise representation, is avoided by using adaptive smoothing techniques. In these algorithms the weights of the smoothing kernel are updated at every iteration to restrict the smoothing to homogeneous regions. Several adaptive methods have been described in the literature (see Saint-Marc and Medioni, 1988, for a review of earlier results). Perona and Malik (1987) proposed an anisotropic diffusion process as a smoothing procedure. Diffusion barriers defined at large values of the gradient keep the smoothing within the boundaries of piecewise constant regions. They proved that no new discontinuities can be introduced during the diffusion process. A simplified anisotropic diffusion technique using similar principles was described by Saint-Marc and Medioni (1988).

The correct choice of the conduction coefficient (defining the strength of the diffusion barrier) is essential for efficient discontinuity preserving smoothing, leading to an increased sensitivity of the results to the values of the design parameters. Many iterations (on the order of hundreds in some cases) are required to achieve the final result. Anisotropic diffusion methods always collapse the input into a piecewise constant

image. To apply the method to more general image structures, Saint-Marc and Medioni (1988) first transformed the input to carry the desired information in a piecewise constant representation. For example, in the case of piecewise linear range images, the image is first differentiated and the resulting first derivative is smoothed. The transformation into a piecewise constant image, however, enhances the effect of the noise and it is not clear if the method can recover the original signal if the signal-to-noise ratio is high.

Somewhat related to adaptive smoothing techniques is the use of non-linear local operators to collapse the image into a piecewise constant representation. The symmetric nearest neighbor filter (Harwood *et al.*, 1987) is a good example of this subclass. The value of the pixel in the center of the window is modified based on pairwise differences of symmetrically located pixels. The method appears to be sensitive to local structure, and produces many relative small constant regions.

A different homogeneity-based approach is the minimal-length encoding technique proposed by Leclerc (1988, 1989) and Pednault (1989) based on information theoretical concepts. The images were modeled as piecewise polynomial regions, corrupted additively with white, Gaussian noise. Local differences in gray level were used as information about possible discontinuities. The segmentation is achieved as the output of a global optimization procedure seeking a description of the image in a minimum number of bits. The minimal-length encoding method tends to extract the largest homogeneous regions and reduce the number of accepted discontinuities. The reliability of the discontinuities is described by a stability measure.

In connection with homogeneity-based methods of segmentation we should also mention the large class of visual reconstruction methods (see Blake, 1989, for a comprehensive list of references). These methods seek a piecewise continuous representation of the input through the minimization of an energy function derived from a Markov random field description of the image structure. The procedures yield the maximum a posteriori probability estimate of the image. It is beyond the scope of the paper to discuss this class of algorithms. We note, however, the importance of "line processes" in the definition of the energy functions. These processes assign a locally computed discontinuity measure to every location.

In the above two sections we surveyed the less "traditional" methods of low-level image analysis. Several common characteristics emerge for each of the two different approaches. We discuss them in the next section.

2.3 Discussion of the Two Approaches

We have already mentioned at the beginning of Section 2.1 that in a piecewise polynomial image the number of possible discontinuity profiles is very large. For example, in one dimension with polynomials having maximum degree $p-1$, if we take into account only

those discontinuities which do not have a step jump when at least one of the two polynomials has $p > 1$, we already obtain $\binom{p+1}{2}$ possible cases. (The number results from the combinations with replacement nature of the problem.) "Traditional" edge detection techniques require smoothing of the image, and therefore the discontinuity profiles are subject to data-dependent distortions, further multiplying the possibilities.

In homogeneity-based approaches, the number of features sought is never more than p , the number of polynomial coefficients. We can also assume that far from discontinuities local smoothing does not significantly modify the order of a homogeneous patch, i.e., after smoothing we can still approximate the patch with a model having the same degree. Another important advantage of seeking homogeneous patches lies in the similarity of the models. The parameters of a constant fit and of a high dimensional polynomial surface fit are computed using the same mathematical technique, regression analysis.

This computational convenience is also used in many of the recent discontinuity-based approaches. To avoid a combinatorial explosion of different discontinuity profiles the residuals of polynomial fits are used. Statistics are then computed in neighborhoods around the location to be analyzed. It is assumed that the detection criterion takes on extreme values when the neighborhoods meet at a discontinuity. Such methods are, however, restricted to one dimension, since in two dimensions a unique mapping between the value of the detection criterion and the local image structure usually does not exist.

The majority of the homogeneity-based approaches discussed in Section 2.2 require the definition of a discontinuity measure at every location in the image. Note that when we define this measure the problem of discontinuity profile diversity is back again! To extract only the significant discontinuities, i.e., to define large homogeneous patches, numerous iterations of the algorithms are required. The reliability of the discontinuity measure is of importance and limits performance in the presence of severe corrupting noise.

Collapsing the input image into a piecewise polynomial description can also be achieved without making explicit use of a discontinuity measure. Computation of the facet model is the best example of such a homogeneity-based technique. Most of the discontinuities are preserved by updating the value of a pixel using the window that provides the best fit. The least squares regression technique employed in the facet model, however, may yield erroneous results close to transitions.

To avoid the effect of discontinuities as much as possible, most homogeneity-based methods use pixel level parallelism. Thus after collapsing the image into the piecewise polynomial description region growing is required to delineate a homogeneous patch. If the description is not very successful region growing may fail.

We conclude that the class of homogeneity-based

methods is more adequate for low-level image analysis since the number of possible models depends only linearly on the assumed polynomial image structure. The drawback of these methods is the need for an initial hypothesis about the discontinuities and the pixel level definition of the output.

In this paper we introduce a new, robust technique to analyze local image structure which is immune to the presence of a discontinuity in the processing window. The Gaussian nature of the corrupting noise can be relaxed; the method handles a large variety of noise processes including impulse noise. The independence of the method from hypotheses on discontinuities allows processing of the image in non-overlapping tessellations, thus loosening the need for pixel level parallelism. Delineation of larger homogeneous patches by region growing is achieved by the same technique.

3 Regression Analysis

For images with piecewise polynomial structures local analysis involves the recovery of the polynomials' coefficients. The problem is known in the statistical literature as regression analysis. In this section we first show that even under ideal processing conditions with no discontinuity present in the local processing window least squares based techniques cannot yield satisfactory results. We then argue for the importance of applying robust regression methods to computer vision problems. We show that of the many robust regression techniques developed in the statistical literature, the least median of squares (LMedS) estimator is best suited for local image analysis. We also give a detailed algorithm for its computation.

3.1 Least Squares Regression in Image Analysis

Least squares regression is the estimation technique most frequently employed in computer vision. The properties of the estimates, however, are rarely given any consideration. In this section we show that most of these estimates are unreliable even in the ideal case in which the assumed model is in agreement with the data. The more realistic case of inconsistency between the model and the data is discussed in the next section.

Let $z_{-n,-n}, \dots, z_{n,n}$ be samples of the noisy image on a square window centered at the origin. Note that we define the image on a grid with nodes corresponding to integers. Assume that the image is a two-dimensional polynomial corrupted by additive noise. That is, assume that we have succeeded in placing our processing window on a homogeneous patch of the image. The data then obeys the model

$$z_{i,j} = \sum_{k=0}^{p-1} \beta_k x_k(i,j) + v_{i,j} \quad i, j = -n, \dots, n \quad (1)$$

where the p regressor variables $x_k(i,j)$ are of the form $i^{k_1} j^{k_2}$ ($k_1, k_2 = 0, 1, \dots$). We prefer this somewhat awkward definition of the polynomial in order to

emphasize the number of regression coefficients β_k that must be estimated. In matrix notation (1) becomes

$$\mathbf{z} = \mathbf{X}\boldsymbol{\beta} + \mathbf{v} \quad (2)$$

where \mathbf{z} is a $(2n+1)^2 \times 1$ vector of data, \mathbf{X} is a $(2n+1)^2 \times p$ matrix of regressor variables, $\boldsymbol{\beta}$ is a $p \times 1$ vector of regression coefficients, and \mathbf{v} is a $(2n+1)^2 \times 1$ noise vector. The noise $v_{i,j}$ has zero mean and is correlated:

$$E[\mathbf{v}] = 0 \quad \text{Cov}[\mathbf{v}] = \sigma^2 \mathbf{V} \quad (3)$$

where \mathbf{V} is a known $(2n+1)^2 \times (2n+1)^2$ covariance matrix.

In the generalized least squares method the regression coefficients are estimated by minimizing the quadratic form

$$(\mathbf{z} - \mathbf{X}\boldsymbol{\beta})' \mathbf{V}^{-1} (\mathbf{z} - \mathbf{X}\boldsymbol{\beta}) \quad (4)$$

where the superscript $'$ means transpose. The minimization yields the estimate vector (see any textbook on linear estimation)

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}' \mathbf{V}^{-1} \mathbf{X})^{-1} \mathbf{X}' \mathbf{V}^{-1} \mathbf{z} \quad (5)$$

having the properties

$$E[\hat{\boldsymbol{\beta}}] = \boldsymbol{\beta} \quad \text{Cov}[\hat{\boldsymbol{\beta}}] = \sigma^2 (\mathbf{X}' \mathbf{V}^{-1} \mathbf{X})^{-1} \quad (6)$$

The estimate $\hat{\boldsymbol{\beta}}$ of the vector of regression coefficients is an unbiased estimate of $\boldsymbol{\beta}$. It can be shown that $\hat{\boldsymbol{\beta}}$ has minimum covariance and thus is the best linear unbiased estimate of $\boldsymbol{\beta}$. Optimal, however, does not necessarily mean satisfactory! If \mathbf{v} is not a Gaussian noise process, all linear estimators return erroneous fits (Hampel *et al.*, 1986, p. 33). If the noise is assumed to be Gaussian the estimate (5) is also identical with the maximum likelihood solution. The covariance matrix $\sigma^2 \mathbf{V}$ of the noise was assumed known. When this information is not available and we mistakenly take $\mathbf{V} = \mathbf{I}$, the identity matrix, the estimator $\hat{\boldsymbol{\beta}}$ remains unbiased but no longer has minimum variance.

We have assumed that the model (1) agrees with the data and have seen that in this case the least squares estimate is the best estimate we can obtain. We show now, however, that this estimate is often not reliable unless the processing window is larger than the ones typically used in computer vision. Let us estimate the coefficients of a planar fit ($p=3$) in the presence of i.i.d. noise ($\text{Cov}[\mathbf{v}] = \sigma^2 \mathbf{I}$). The fit is then of the form

$$\hat{z} = \hat{\beta}_0 + \hat{\beta}_1 i + \hat{\beta}_2 j \quad i, j = -n, \dots, n \quad (7)$$

and it is immediate to obtain from (6)

$$\text{Cov}[\hat{\boldsymbol{\beta}}] = \sigma^2 \begin{bmatrix} \frac{1}{(2n+1)^2} & 0 & 0 \\ 0 & \frac{3}{n(n+1)(2n+1)^2} & 0 \\ 0 & 0 & \frac{3}{n(n+1)(2n+1)^2} \end{bmatrix} \quad (8)$$

The estimates of the three parameters, the intercept

and the two slopes, are independent as one should expect. A simple quality measure for unbiased estimates is the ratio between the correct value β_k and the standard deviation of the estimate. We obtain

$$\eta_0 = (2n+1) \frac{\beta_0}{\sigma} \quad \eta_1 = (2n+1) \frac{\beta_1}{\sigma \sqrt{3}} \quad (9)$$

The larger the ratio the better the estimate. We will consider an estimate reliable if η is larger than 5. This roughly corresponds to $\pm 10\%$ estimation error. The more accurate measure of confidence interval involves additional assumptions and for our purposes (9) suffices. From the expression for η_1 we can obtain the normalized upper bound for the standard deviation of the noise yielding reliable estimates:

$$\frac{\sigma}{\beta_1} \leq \frac{(2n+1) \sqrt{n(n+1)}}{5\sqrt{3}} \quad (10)$$

In Table 1 the values of this bound are shown for several window sizes.

Table 1: Dependence of the normalized noise upper bound on window size.

n	Window Size	Upper Bound
1	3×3	0.49
2	5×5	1.41
3	7×7	2.80
4	9×9	4.65
5	11×11	6.96
6	13×13	9.73

In images the slope values β_1 have the order of magnitude of units. The bounds in Table 1 show that small windows cannot provide reliable estimates for the slopes of a planar fit even in the presence of negligible noise. For example if $\beta_1 = 1$ a 5×5 window is able to handle only noise with $\sigma \leq 1.41$! The intercept estimate is much more reliable since β_0 has the order of magnitude of hundreds and η_0 is always larger than 5.

We conclude that the small window based least squares operators usually employed in computer vision cannot return a reliable estimate of a polynomial surface, except for its intercept. Since the least squares estimate is the best estimate with a linear estimator structure, the only way to recover the correct polynomial fit is by employing larger processing windows. Large windows, however, with high probability contain discontinuities, i.e., the data no longer can be represented by a single polynomial surface as in (1). In this case the least squares technique is no longer optimum. We have assumed that the window is centered on a homogeneous patch, a hypothesis which cannot be satisfied in real applications. When no a priori information about discontinuities is available, many windows will cover piecewise polynomial local image structures. We now introduce the concept of breakdown point which captures the property required of an estimator in order for it to be able to handle

piecewise polynomial models.

3.2 Characteristics of a Regression Method

For convenience, the discussion in the following sections deals with one-dimensional signals, but it can readily be extended to higher dimensions. The residual r_i is defined as the difference between the data and the value of the estimated fit \hat{z}_i :

$$r_i \triangleq z_i - \hat{z}_i = z_i - \sum_{k=0}^{p-1} \hat{\beta}_k i^k \quad i = -n, \dots, n \quad (11)$$

Note that in spite of (1), r_i does not necessarily have the same probability law as v_i since $\hat{\beta}_k$ depends on the data z_i . The regression coefficient estimates are obtained by minimizing a penalty function of the residuals r_i . A large variety of penalty functions are employed in the statistical literature. An important class is that of the continuous, symmetric, positive valued penalty functions $\rho(u)$ with a unique minimum at $u = 0$. For this class, finding the regression coefficients is equivalent to the minimization problem

$$\min_{\hat{\beta}} \sum_{i=-n}^n \rho(r_i) \quad (12)$$

The particular case $\rho(u) = u^2$ yields the least squares regression discussed in Section 3.1 for the case $\mathbf{V} = \mathbf{I}$. The noise is assumed zero mean in the least squares model (3). This is, however, not necessarily true for the local window operators used in computer vision. Small data sets often yield significantly skewed samples, even when the original noise process is symmetric (Rousseeuw and Leroy, 1987, p. 166) and thus asymmetric noise can also be generated by a zero mean random process.

The limitation of least squares regression to data that arise from a polynomial surface is related to its inability to handle non-zero mean noise. Whenever two surfaces meet within the processing window we can regard the surface generating the majority of the pixels as the signal and the remaining pixels as samples corrupted by noise. In this case the noise always has non-zero mean.

In the last two decades, statisticians have developed new, robust regression techniques which tolerate deviations from the assumed models. The books of Huber (1981), Hampel *et al.* (1986) and Rousseeuw and Leroy (1987) are the leading references for robust statistics. Robust methods make use of a priori available information through the assumed model from which, however, deviations are allowed. Thus robust methods are more powerful than non-parametric methods which, in order to avoid erroneous assumptions, do not involve models at all. See Hampel *et al.* (1986), Section 1.1b, for a discussion of the place of robust methods among other statistical techniques.

A large variety of robust methods exist today. To evaluate which of these techniques is best suited for image analysis, we discuss three characteristics of regression methods. The *one-dimensional location*

estimation problem is used as an example to illustrate the meanings of different regression characteristics. This is the simplest regression problem, in which the value of the best fitting constant $\hat{\beta}_0$ to the data z_i , $i = -n, \dots, n$ is sought. Two penalty functions $\rho(u)$ will be considered for (12). In the case of least squares regression, also known as L_2 regression, the solution of the location estimation problem is the arithmetic mean

$$L_2: \hat{\beta}_0 = \frac{1}{2n+1} \sum_{i=-n}^n z_i. \quad (13)$$

When $\rho(u) = |u|$ is used in (12), the least absolute values or L_1 regression is obtained. The solution is the median

$$L_1: \hat{\beta}_0 = \text{med } z_i. \quad (14)$$

The *relative efficiency* of a regression method describes the quality of the estimate by a positive number between 0 and 1. It is computed as the ratio between the lowest achievable variance for the regression coefficients and the variance obtained when the current method is employed. In the former case any method is allowed. The asymptotic relative efficiency is obtained when the size of the data set n tends to infinity. The relative efficiency clearly depends on the nature of the corrupting noise. In most cases the noise is assumed to be normal.

In our examples, the asymptotic relative efficiencies for Gaussian corrupting noise are

$$\text{mean: } 1 \quad \text{median: } \frac{2}{\pi} = 0.637. \quad (15)$$

The arithmetic mean (13) is indeed a minimum variance estimator, achieving the lowest possible (Cramer-Rao) bound. The relative efficiency of the median is lower; for the same degree of precision a sample size $\pi/2$ larger should be employed (Wilks 1962, p. 364). This is due to the fact that while L_2 regression is optimal for Gaussians, L_1 regression is optimal for Laplace (doubly exponential) noise processes.

The convergence rate of the estimators is also studied in statistics—that is, as the size of the data increases, how fast the ratio of variances approaches the asymptotic relative efficiency. Both L_2 and L_1 regression converge toward zero at a rate of $n^{-1/2}$.

The *complexity* of a regression method refers to the amount of computation required to obtain the estimate. It is desired to keep the amount of computation low, although with progress in parallel computer hardware, the precise definition of what low means is no longer clear. In our examples we have

$$\text{mean: } O[n] \quad \text{median: } O[n] \quad (O[n \log n]) \quad (16)$$

so that the complexity depends linearly on the data size. The $O[n]$ algorithm for the computation of the median is not recommended if $n < 50$ (Aho *et al.* 1974, Section 3.6). Thus for the local operators used in computer vision, the $O[n \log n]$ complexity median algorithm involving direct sorting is more appropriate.

The *breakdown point*, ϵ^* , of a regression method is the smallest fraction of contamination of the data which yields arbitrarily incorrect estimates. This is the most important regression characteristic for our discussion since it captures the behavior of the estimator when the data severely deviates from the model. The data points yielding severe deviations are called *outliers*. Note that whenever two surfaces meet in a processing window (i.e., a discontinuity is present) the pixels belonging to one of the surfaces can be regarded as outliers for the fit to the other one. The largest possible value for ϵ^* is 0.5, since above this bound the outliers become the majority. Both finite-sample and asymptotic definitions exist for the breakdown point. In our examples

$$\text{mean: } \frac{1}{2n+1} \rightarrow 0 \quad \text{median: } \frac{n+1}{2n+1} \rightarrow 0.5 \quad (17)$$

where the numerical values are the asymptotic breakdown points. One large, incorrect value suffices to corrupt the arithmetic mean, while the median tolerates half the data being corrupted.

We cannot conclude, however, that in the general case L_1 regressions have $\epsilon^* = 0.5$. When the data is no longer defined in a compact region on a lattice, i.e., some points could lie far from the rest, it can be shown that the asymptotic breakdown point of L_1 regression is 0 (Rousseeuw and Leroy, 1987, p. 20 and p. 145). A similar remark is valid for minimax (or L_∞) regression in which the minimization of the maximum squared residual is employed to compute the regression coefficients (*ibid.*, p. 125). These remarks are of significance for computer vision. It suggests that L_1 and L_∞ regression methods should not be applied to problems with sparse data (e.g. line fitting to noisy edge detector output) since distant points could yield incorrect results.

The equivariance properties of an estimator are of lesser importance for us. They characterize the behavior of the estimator under transformations of the regressor variables or of the data. (For more detailed definitions see Hampel *et al.* 1986, Section 5.3; Rousseeuw and Leroy, 1987, p. 116.)

Although they represent a class of robust estimators, we are not interested here in techniques based on order statistics (trimmed-mean filters, for example), which have already been employed for a long time in computer vision (see Coyle *et al.* 1989, for a complete literature survey). Recently, estimators making use of more sophisticated robust regression methods have been designed. *M-estimators*, corresponding to different $\rho(u)$ functions in (12), are a family of robust techniques. In the broadest sense our previous two examples are M-estimators, but usually M-estimators use more complex $\rho(u)$ functions to limit the influence of outliers on the regression coefficients. The books of Huber (1981) and Hampel *et al.* (1986) offer a complete treatment of the subject. The minimization problem (12) is solved for M-estimators as iteratively reweighted least squares with the definition of the weights depending on $\rho(u)$. Note that the resulting

estimators no longer have linear structure and thus superior estimates can be obtained for non-Gaussian corrupting noise. The reliability of the initial guess used in the iterative process is extremely important; otherwise the final results could be nonrobust (Hampel *et al.*, 1986, p. 106 and p. 263). Since least squares is employed, the relative efficiency of M-estimators is good and the complexity is low.

The attractiveness of the good computational features of M-estimators is somewhat lessened by their low breakdown points. It can be shown that no M-estimator can have a breakdown point larger than $\epsilon^* = \frac{1}{p}$, where p is the dimensionality of the parameter space (Hampel *et al.*, 1986, p. 329). For example, in the case of biquadratic surface fitting ($p = 6$) the upper bound is $\epsilon^* < 0.167$, i.e., only about 15% of the data can be severely corrupted if we want to be able to recover the correct fit. Note that the breakdown point is a "worst-case" characteristic. Higher levels of contamination may be tolerated in certain cases, but there is no guarantee that the performance can be repeated for *all* the outcomes of the noise.

Recently M-estimators were applied to several important computer vision problems. Kashyap and Eom (1988) employed them in edge detection tasks, to estimate the parameters of causal autoregressive image models driven by Gaussian noise contaminated with a small fraction of impulse noise. The same technique was used by Koivo and Kim (1989) for classification of surface defects on wood boards. Haralick and Joo (1988) used M-estimators for pose estimation. A similar algorithm was used by Lee *et al.* (1989) for estimating 3D motion parameters. Besl *et al.* (1988) implemented a hierarchical surface fitting scheme based on M-estimators, in which successively higher degree fits (up to bicubic) are evaluated through a fit quality measure.

High breakdown point regression methods are desirable in computer vision. Let us restrict ourselves for the moment to an uncorrupted one-dimensional example having piecewise polynomial structure of maximum degree $p-1$. Assume that the regression technique used has $\epsilon^* = 0.5$ and the operator returns the estimated value of the pixel in the window center, located in the $(n+1)^{\text{th}}$ position. Since the input is an ideal piecewise polynomial signal, a contiguous group of $n+1$ pixels always carries information about the same polynomial. The 0.5 breakdown point assures that the estimated regression coefficients are those of this polynomial and the value returned by the operator is the value of the polynomial at the window center. The noiseless one-dimensional piecewise polynomial signal remains undistorted when processed by a regression operator with breakdown point $\epsilon^* = 0.5$. This *root-signal* property is not necessarily valid in two dimensions. When the window is centered on a corner, it is not always the case that 51 percent of the pixels belong to the surface in the window center.

Several robust regression methods with ϵ^* close to 0.5 exist (see the book of Rousseeuw and Leroy, 1987).

In the next section we show that for low-level processing in computer vision the *least median of squares* (LMedS) regression estimator appears to be the most appropriate. (We prefer to use the LMedS notation instead of LMS, which is used in the statistical literature, to avoid confusion with the usage in the image processing field where LMS stands for least mean squares.) We have already applied this technique to clustering problems and to recovery of images corrupted by impulse noise (Kim *et al.* 1989, Jolion *et al.* 1990). Tirumalai and Schunck (1988) and Sinha and Schunck (1989) used least median of squares to interpolate sparse data over a rectangular lattice, and Kumar and Hanson (1989) used it to solve the pose estimation problem.

3.3 Least Median of Squares Regression

Rousseeuw (1984) introduced the least median of squares (LMedS) regression method to the statistical literature, although the idea was mentioned by Hampel ten years earlier. If not otherwise specified, all the references in this section refer to page numbers in the book of Rousseeuw and Leroy (1987). We prefer to use this excellent, application oriented source, instead of the original papers which approach the issues on a more general level.

In the least median of squares technique the following minimization problem must be solved to estimate the regression coefficients:

$$\min_p \text{med}_i r_i^2. \quad (18)$$

Note that (18) is of a different nature than (12), where an analytical expression in the residuals had to be minimized. The objective function in (18) can have at most $O[n^p]$ local minima (*ibid.*, p. 206) and there always exists a solution to the minimization problem (*ibid.*, p. 113). It can be shown that (18) is computationally preferable to the apparently equivalent least median of absolute values (*ibid.*, p. 170).

The breakdown point of least median of squares regression is (*ibid.*, p. 118)

$$\epsilon^* = \frac{n-p+2}{2n+1} \rightarrow 0.5. \quad (19)$$

The upper bound on any regression equivariant estimator (*ibid.*, p. 124)

$$\epsilon^* = \frac{[(2n-p+1)/2] + 1}{2n+1} \quad (20)$$

(where $[\cdot]$ is the integer function) is slightly larger than (19). This bound can be achieved by the repeated median method (which is not affine equivariant and is computationally prohibitive) and by variants of LMedS (which are computationally more expensive than LMedS). From (19) we can see that if $n+p$ data points are always uncorrupted in the processing window, the root-signal property discussed at the end of Section 3.2 is satisfied.

The efficiency of least median of squares regression is low since it involves median operations. Its convergence rate is only $n^{-1/3}$ (*ibid.*, p. 178). This

disadvantage can be compensated by applying a one-step reweighted least squares (RLS) procedure to the output of the LMedS estimator. We now proceed to describe the sequence of computations yielding the least median of squares estimates.

The objective function (18) is not an analytical expression in the residuals and the regression coefficients. The least median of squares estimator uses a *projection pursuit* type procedure (*ibid.*, p. 143) in which the data set is projected onto a one-dimensional space.

Let a p -tuple of distinct data points be characterized by the indices

$$\mathbf{j} = \{i_1, \dots, i_p\} \quad \mathbf{j} = 1, 2, \dots, \binom{2n+1}{p}. \quad (21)$$

For every p -tuple the values of the initial estimate regression coefficients

$$\hat{\beta}_k(\mathbf{j}) \quad k = 1, \dots, p-1 \quad (22)$$

are computed by least squares regression. Since the number of data points and the number of unknowns are equal the regression coefficients can be computed from closed form expressions. Note that the coefficient $\hat{\beta}_0(\mathbf{j})$ is not sought. The data is then projected onto the β_0 subspace by computing

$$\alpha_i(\mathbf{j}) \triangleq z_i - \sum_{k=1}^{p-1} \hat{\beta}_k(\mathbf{j}) i^k \quad i = -n, \dots, n. \quad (23)$$

The next step is to determine the mode (location of the maximum) of the probability distribution of $\alpha(\mathbf{j})$. The $2n+1$ values are sorted in ascending order

$$\alpha_{[1]}(\mathbf{j}) \leq \alpha_{[2]}(\mathbf{j}) \leq \dots \leq \alpha_{[2n+1]}(\mathbf{j}) \quad (24)$$

and the differences

$$\gamma_l(\mathbf{j}) \triangleq \frac{\alpha_{[n+l]}(\mathbf{j}) - \alpha_{[l]}(\mathbf{j})}{2} \quad l = 1, 2, \dots, n+1 \quad (25)$$

are defined. The mode is then located at $l = m_j$ yielding

$$\gamma_{m_j}(\mathbf{j}) = \min_l \gamma_l(\mathbf{j}). \quad (26)$$

The value of the mode is taken as the value of the yet undetermined intercept for the given p -tuple

$$\hat{\beta}_0(\mathbf{j}) = \frac{\alpha_{[n+m_j]}(\mathbf{j}) + \alpha_{[m_j]}(\mathbf{j})}{2} \quad (27)$$

The ordering relations

$$(\hat{\beta}_0(\mathbf{j}) - \alpha_{[i]}(\mathbf{j}))^2 \begin{cases} \leq \gamma_{m_j}^2(\mathbf{j}) & \text{if } m_j < i < m_j + n \\ = \gamma_{m_j}^2(\mathbf{j}) & \text{if } i = m_j \text{ or } i = m_j + n \\ \geq \gamma_{m_j}^2(\mathbf{j}) & \text{if } 1 < i < m_j \text{ or } m_j + n < i \leq 2n+1 \end{cases} \quad (28)$$

are immediate to verify. The relation (28) can also be written as

$$\gamma_{m_j}^2(\mathbf{j}) = \text{med}_i (\hat{\beta}_0(\mathbf{j}) - \alpha_i(\mathbf{j}))^2 = \text{med}_i r_i^2(\mathbf{j}). \quad (29)$$

For a more rigorous proof, and comments on the need

to use squared and not absolute values in (28), see *ibid.*, p. 166. If more than one minimum is found in (26), the value of the mode is taken as the average of the midpoints of the intervals. The minimization problem defining the LMedS regression coefficients (18) is now reduced to finding

$$\gamma^2 \triangleq \gamma_d^2 = \min_{\mathbf{j}} \gamma_{m_j}^2(\mathbf{j}). \quad (30)$$

The final regression coefficient LMedS estimates are obtained from the p -tuple achieving the minimum in (30), $\hat{\beta}_k(\mathbf{d})$.

The way LMedS estimates are computed also provides a physical interpretation of the least median of squares method (*ibid.*, p. 126). First the thinnest hyperstrip containing half the data points is determined. The width is measured along the dimension of β_0 , i.e., of the intercept. The LMedS surface fit is then the medial axis of the hyperstrip.

The global minimum value, γ (30), also returned by the LMedS estimator can be employed for a robust estimate of the standard deviation

$$\hat{\sigma} = 1.4826 \left(1 + \frac{5}{2n-p+1} \right) \gamma \quad (31)$$

where the term $5/(2n-p+1)$ is the finite sample size correction (*ibid.*, p. 202) and 1.4826 is the correction factor for median based standard deviation estimates. This correction factor is based on normal distribution around the fit. This assumption is satisfactory in most cases, but if the nature of the noise is a priori known other correction factors can easily be computed. For example, the uniform distribution yields 1.1547. The residuals of the LMedS fit normalized by the standard deviation (a scale estimate) define the set of weights w_i :

$$w_i = \begin{cases} 1 & \frac{|r_i|}{\hat{\sigma}} \leq 2.5 \\ 0 & \frac{|r_i|}{\hat{\sigma}} > 2.5 \end{cases} \quad (32)$$

All the data points having $w_i = 0$ are outliers, i.e., severe contaminations of the data. The LMedS estimator tolerates a discontinuity in the processing window. If at least half the pixels belong to one surface the most of the remaining pixels are detected as outliers. The quality of the procedure depends on the presence of noise corrupting all the pixels and on the underlying image structure as will be discussed in the next section.

The LMedS regression algorithm requires an extremely large amount of computation. From (21) we can see that the number of p -tuples is $O[n^p]$ and for every p -tuple the mode seeking procedure has complexity $O[n \log n]$. Edelsbrunner and Souvaine (1988) proposed an algorithm of $O[n^2]$ for LMedS based linear regression ($p = 2$), but its practicality in applications especially in higher dimensions is not clear.

In order to reduce the otherwise impractical

amount of computation of the LMedS estimator only a reduced set of p -tuples are selected at random from the total $\binom{2n+1}{p}$ (Rousseeuw and Leroy, 1987, p. 198). By doing so, we accept a probability of error Q in recovering the correct regression coefficients. Let the fraction of outliers in the data be $0 \leq \epsilon < 0.5$. The probability that from q independently chosen p -tuples at least one does not contain any outliers is

$$1 - [1 - (1 - \epsilon)^p]^q \leq 1 - Q. \quad (33)$$

From (33) we can find the value of q for any given p, ϵ and Q . For example, if in planar surface fitting ($p = 3$) the data contains $\epsilon = 0.45$ outliers and the tolerated probability of error is $Q = 0.01$, only 26 triples must be used, independent of the processing window size. Rousseeuw and Leroy (p. 199) recommend taking a larger set of p -tuples than the number obtained from (33). Note that precautions must be taken to assure that no two p -tuples are the same, a possible outcome if the size of the processing window is small. We avoided such ties by using a novel random sampling algorithm (Mintz and Amir, 1989).

In Section 3.1 we have shown that the reliability of least squares estimates is extremely low when computed based only on a few samples. Can we improve the performance of the LMedS estimator if instead of p -tuples (and closed formulas) we use larger samples (and explicit least squares)? From (33) we see that for the same probability of error Q the number of tuples increases exponentially with the size of the tuple. If instead of 3-tuples, 5-tuples are used in the above example, at least 90 distinct quintuples must be drawn. On the other hand, the standard deviation of the least square estimates decreases only linearly with the size of the tuple and remains large for all the feasible tuple sizes. We conclude (based also on simulation results) that no real improvement can be obtained by increasing the size of the tuples while the amount of required computation increases drastically.

The above described random sampling procedure is similar to the RANSAC paradigm of Fischler and Bolles (1981). In RANSAC, however, the model derived from the first p -tuple chosen determines the set of data points agreeing with it within a given tolerance limit. If the set is large enough, no more p -tuples are drawn. The only significant difference between RANSAC and LMedS is that the latter generates the error measure, while the former must be supplied with it. For a more detailed discussion see Meer *et al.* (1990).

We conclude this section with a short description of the reweighted least squares (RLS) postprocessing step. Rousseeuw and Leroy (1987, p. 211) show that using reweighted least squares after LMedS is more effective than using a one-step M-estimator. In computer vision the application of this postprocessing step should be examined carefully; we discuss it in the next section.

The weights w_i allow computation of a new standard deviation estimate by applying the well known formula for all the data points which are not outliers:

$$\hat{\sigma}^* = \sqrt{\frac{\sum_{i=-n}^n w_i r_i^2}{\left[\sum_{i=-n}^n w_i - p\right]}}. \quad (34)$$

New weights w_i^* can be determined by using $\hat{\sigma}^*$ in (32). Let \mathbf{w}^* be the $(2n+1) \times 1$ vector of these weights. If we define $\mathbf{V}^{-1} = (\mathbf{w}^*)(\mathbf{w}^*)'$ the reweighted least squares regression coefficients are given by (5). This solution is ordinary least squares applied to the inliers of the window. The standard deviation value can be updated by repeating (32) and (34) for the resulting fit values.

4 Least Median of Squares Estimators in Image Analysis

In computer vision the optimum breakdown point is the most important property of the least median of squares (LMedS) estimators. It allows analysis of images near discontinuities by recovering the surface which contains the majority of pixels in the processing window. The local structure then can be described in an inlier/outlier representation. The LMedS estimator, however, was designed to suit the data used in statistics. In this section we show the limitations of direct (one-step) application of LMedS estimators to images. We then propose an algorithm based on LMedS computational modules that eliminates most of the inherent problems of the original estimator.

4.1 Limitations of LMedS Estimators for Image Data

Analysis of the LMedS estimator computation (see Section 3.3) shows that the necessary and sufficient condition to obtain the correct estimate is the existence of at least one p -tuple carrying the true regression coefficients of the polynomial surface defined by the majority of the pixels. Noise corrupting *all* the samples can only be tolerated if it does not yield significant degradation of *all* the initial estimates used in the projection pursuit procedure. In statistics applications this condition is satisfied since the data usually contains just a few outliers caused by severe measurement errors or faulty inputting.

In computer vision ideal conditions for LMedS estimators are only rarely met. Near transitions about half the pixels are outliers; the noise corrupting all the pixels thus destroys the dichotomy between the inliers and outliers. The high breakdown point loses its meaning; a p -tuple chosen from the correct surface does not necessarily yield a reliable estimate.

Nevertheless, there exist applications in which LMedS estimators can be used without additional safeguards. Such an application is the interpolation of noiseless (or weakly corrupted) data. That is, we want to recover from a sparse set of pixels an image defined on a regular grid. For our discussion neither the problem of data density required for satisfactory reconstruction, nor the influence of noise due to the presence of image structure corresponding to polynomials with degrees higher than $p-1$, is of importance.

In a simple LMedS based non-linear interpolation procedure a $(2n+1) \times (2n+1)$ window is centered at every pixel and the intercept value of the fit is allocated to it. Thus the pixel gets its value from the fit containing the majority of the available samples in the window. The sparseness of the data and the minimum recoverable feature size define the trade-offs on the window size. From the 128×128 aerial image in Figure 1a 14% of the pixels were removed by random selection (Figure 1b). A 5×5 window based first order LMedS estimator was then applied at every pixel to interpolate the image (Figure 1c). The larger features are correctly recovered. The smaller features can be recovered only if the available samples are the majority in the windows centered on them.



a) Original aerial image.

Figure 1: Least median of squares as a non-linear interpolation operator.

When a relatively small number of pixels in an image are corrupted by asymmetric (non-zero mean) noise processes, an LMedS based window operator can recover most of the original image. The synthetic image in Figure 2a was corrupted by a Weibull noise process affecting 14% of the pixels (Figure 2b). The process has the probability density function

$$f(u) = \begin{cases} 0 & u < 0 \\ 2ue^{-u^2} & u \geq 0 \end{cases} \quad (35)$$

with mean $\sqrt{\pi}/2 = 0.886$ and variance $1 - \pi/4 = 0.215$. In Figure 2b the noise is multiplied by 50 before being added to the selected pixels. The image recovered with a 5×5 degree-1 LMedS estimator is shown in Figure 2c. Note that except near discontinuities where the number of incorrect samples may now exceed half the window size the estimator recovers the fit. As was mentioned at the end of Section 3.2 the LMedS



b) Image with 14% of the pixels removed.



c) Interpolated image using a 5×5 LMedS operator.

window operator makes systematic errors at corners. In Figure 2d the result of processing the noiseless image (Figure 2a) is shown. At corners the center pixel may belong to the surface containing the minority of the pixels and therefore is assigned to the incorrect fit. A possible way to eliminate this artifact is by using a multiwindow approach similar to the one employed for computation of the facet model (Haralick and Watson, 1981). Fits from several overlapping windows containing the pixel as an inlier are compared and the final fit is taken from the window which has the smallest robust variance estimate. The artifacts at

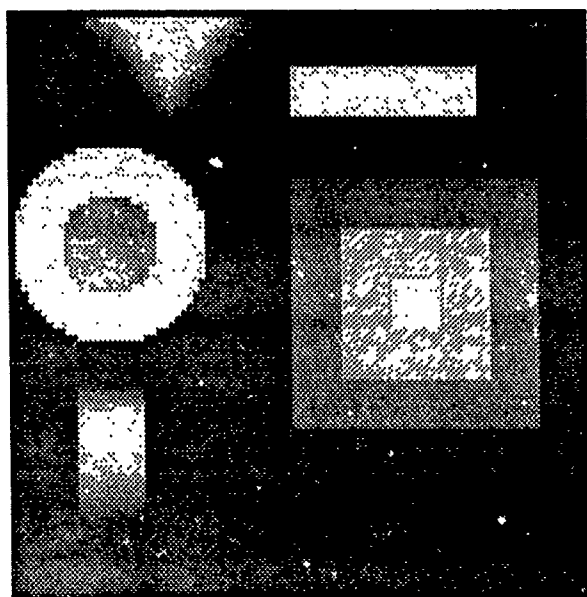
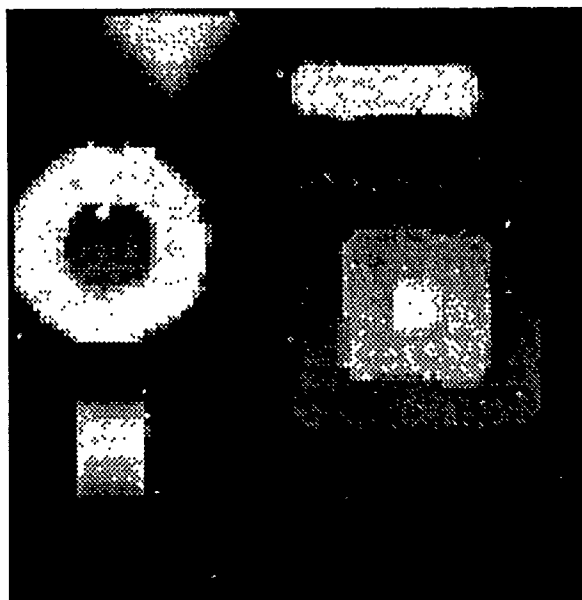
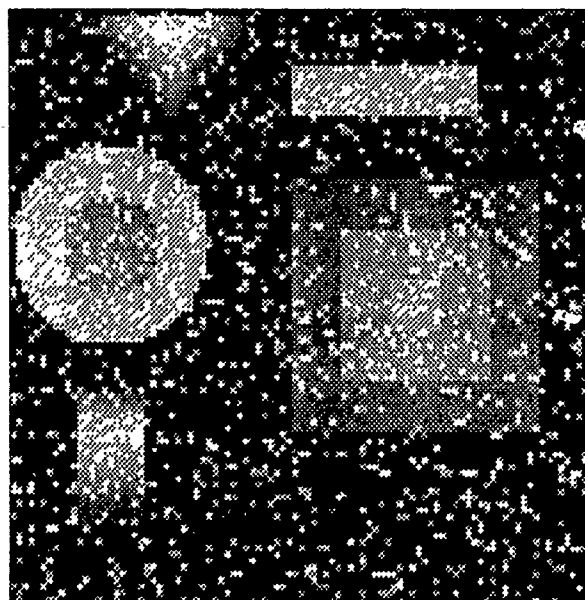


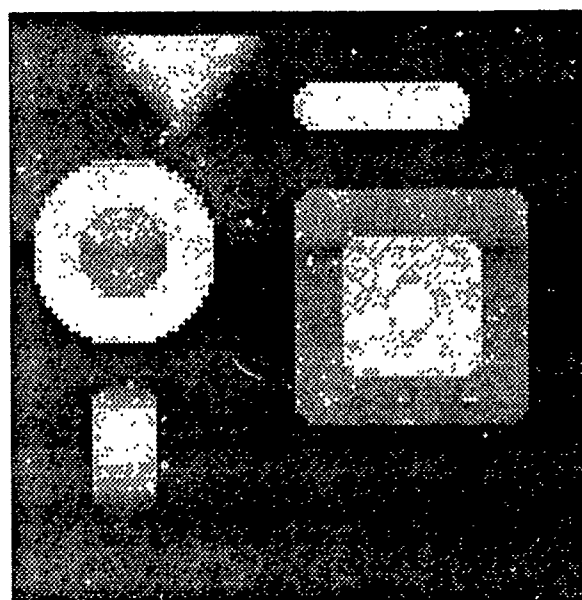
Figure 2a) A piecewise planar synthetic image.



2c) Image filtered by a 5×5 LMedS operator.



2b) Image with 14% of the pixels corrupted by Weibull noise.



2d) The noiseless image processed by a 5×5 LMedS operator.

corners can also be eliminated in other ways as will be discussed in the next section.

In the above examples only a subset of the pixels was corrupted by a noise process, yielding outliers in most cases. In images, however, usually a zero mean additive noise process is present at every pixel. The p -tuple based initial estimates then have large variance and become unreliable. This is especially severe near discontinuities where the number of outliers is close to half the window size and only a few p -tuples are drawn from the surface to be recovered. We now analyze the behavior of LMedS estimators near step

and roof discontinuities in the presence of significant zero mean noise corrupting all the samples. In this paper we will not consider image structures containing higher order polynomial surfaces.

In Figure 3a a noisy step discontinuity with amplitude h is shown. The image is corrupted by noise taking values in the interval $(-a, a)$. This assumption is for convenience only and has no influence on the results. For example, in the case of Gaussian noise we can take $a = 2.5\sigma$. An LMedS based window operator is centered on the edge, i.e. the data has ϵ close to 0.5. For the moment, to recover the horizontal plane

containing the majority of pixels, let us use the degree-0 LMedS estimator ($p = 1$). As was shown in Section 3.3 the zero order estimate is the mode of the pixel values' distribution. Assume that the correct fit is recovered with a negligible error. To detect the outliers in the processing window the standard deviation estimate $\hat{\sigma}$ is also needed (32). This estimate is proportional to the median of the absolute values of the residuals relative to the final fit (31). Our fit is correct and therefore the absolute values of the residuals are distributed as follows: slightly more than half between zero and a , and the rest in the range $(h-a, h+a)$. The value of the median is close to a and $\hat{\sigma}$ is always an overestimate for the noise's standard deviation.

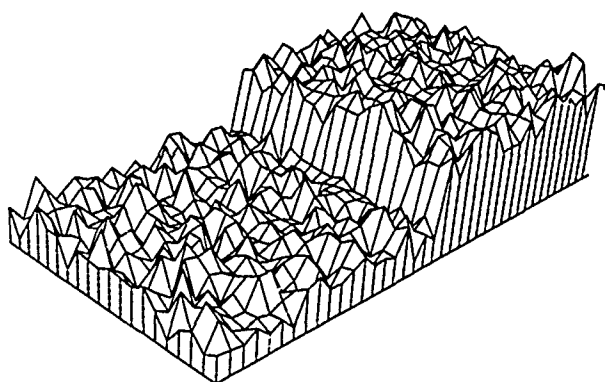
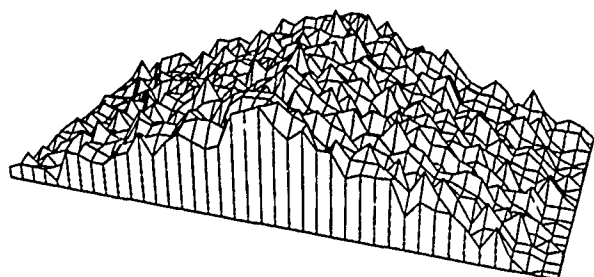


Figure 3a) Noisy step edge.



3b) Noisy roof edge.

We performed 100 trials on a 40 gray level vertical step edge (128/168) corrupted with Gaussian noise, $\sigma = 10$. The LMedS estimator was based on a 5×5 window containing three columns of data derived from the surface with amplitude 128. Thus ten outliers are present in the noiseless window and $\epsilon = 0.4$. The results are shown in Table 2.

Table 2: Results of the degree-0 LMedS estimator applied to a step edge.

Estimate	Min	Max	Mean	Std. dev.
$\hat{\beta}_0$	119	150	128.68	5.14
$\hat{\sigma}$	9.85	29.56	19.92	4.05

We see from Table 2 that the correct fit is recovered with reasonable accuracy but the estimated standard

deviation is about twice its real value. The overestimated standard deviation yields fewer outliers since $2.5\hat{\sigma}$ is already larger than the step size, and local image structure is not identified correctly.

We can also apply a degree-1 LMedS estimator ($p = 3$) to the step discontinuity. As was discussed in Section 3.1, the initial regression coefficient estimates have the correct mean but due to their huge variance can take any value. Assume that we have chosen two 3-tuples, one generating (by chance) the correct horizontal fit, the other yielding a plane tilted at about 45 degrees across the edge. The latter fit is similar to what a least squares estimator would have recovered from the window.

If significant noise is present (a is close to $h/2$), most of the absolute values of the residuals for the tilted plane fit are between 0 and a . The median of this residual sequence is significantly lower than the median of the sequence obtained when the horizontal fit is employed. The minimization criterion of LMedS estimators seeks minimum median residuals, and therefore the tilted plane is preferred. Note that tilted planes are always obtained from 3-tuples since any sample containing pixels from both surfaces can yield one. The estimator will choose the correct, horizontal plane only when the amplitude of the corrupting noise is very small relative to the amplitude of the step. In this case, the tilted plane yields larger residuals. The correct fit, however, implies an overestimate of σ as was shown above.

Results obtained with a degree-1 LMedS estimator applied to the same input data as above, are shown in Table 3.

Table 3: Results of the degree-1 LMedS estimator applied to a step edge.

Estimate	Min	Max	Mean	Std. dev.
$\hat{\beta}_0$	80	160	121.22	13.39
$\hat{\beta}_1$	-10.83	20.00	10.83	5.00
$\hat{\beta}_2$	-12.00	13.83	-0.05	4.52
$\hat{\sigma}$	5.00	18.88	11.31	2.87

The spread of the regression coefficients is very large and any slope value could result. Note that the plane is always tilted across the edge; the mean orientation along that direction is 10.83. The tilted planes yield significantly smaller standard deviation estimates than the correct, horizontal surface fit (see Table 2).

The processing of roof discontinuities (Figure 3b) presents different problems. For the same noise amplitude, the local signal-to-noise ratio (the one of importance in parallel processing) is much smaller at a discontinuity than at a step edge. Small amount of noise already erases the precise location of the intersection of the two planes. The operators are more likely to return a tilted fit when processing this region cannot be eliminated by employing operators with large supports. Assume that the correct fit is

containing the majority of the pixels as well as the correct standard deviation for the noise are recovered. The definition of an outlier implies a minimum deviation from the estimated fit (31). Thus several pixels adjacent to the roof discontinuity but belonging to the other plane are always incorporated into the fit. This fact also reduces the usefulness of reweighted least squares postprocessing in image analysis.

The degree-1 LMedS estimator applied to a single noisy plane underestimates the standard deviation of the noise. A local planar fit aligns the noisy data better than the original plane and thus the distribution of the absolute values of the residuals is shifted downward.

We conclude from the above discussion that several precautions should be taken when the LMedS estimator is used for the analysis of piecewise polynomial image structures:

- The fit estimate should be taken from the estimator using the correct degree for the polynomial surface containing the majority of the pixels in the window. Failing to do so may lead to an erroneous decision, as in the application of the degree-1 LMedS estimator to a noisy step edge.
- The noise's standard deviation should not be estimated simultaneously with the fit. Usually under- or overestimates are obtained, diminishing the efficiency of the outlier detection procedure.
- Often the local operator cannot achieve the correct decision and cooperation between windows should be considered. For example, roof edges are not locally recoverable, but the intersection of surfaces estimated from homogeneous patches could yield the desired result.

Applying a local operator at every pixel does not explicitly define patches resembling polynomial surfaces in the output image. To recover small homogeneous patches in the image, we analyze the output of the pixel level local LMedS estimators with a second LMedS estimation process defined in windows yielding a nonoverlapping tessellation of the input. It is shown in Section 4.2 that this two-step approach also solves most of the above mentioned problems of the LMedS estimators. To delineate larger homogeneous regions the small patches belonging to the same surface must be fused by region growing. When the input image is noisy the structure of the output is not perfectly piecewise polynomial of degree $p-1$. The region growing procedure may stop too early, or may fuse two surfaces with similar parameters. We show in Section 4.3 how the high breakdown point of the LMedS estimator allows its use in a robust region growing algorithm.

4.2 An LMedS Based Image Analysis Algorithm

The goal of image analysis in the piecewise polynomial domain is to identify the parameters of the underlying polynomial surface β_i and the standard deviation of

the noise $\hat{\sigma}$. The inlier/outlier structure within the processing window can then be determined. To avoid the problems discussed in the previous section, we combine an LMedS estimator based on a small window centered at every pixel with a second LMedS estimator operating on a coarser tessellation of the image. The first estimator achieves a robust presmoothing of the image as well as supplying the initial regression coefficient estimates; the second estimator then recovers the underlying polynomial structure.

A small $(2n+1) \times (2n+1)$ window based LMedS estimator is applied centered at every pixel of the image. The estimator is employed as described in Section 3.3 and returns the estimates $\hat{\beta}_i, i = 0, \dots, p-1$ and $\hat{\sigma}$. Since these estimates are available at every pixel they can define several "estimate images". The $\hat{\beta}_0$ -image is of special interest. In Section 3.1 we showed that the intercept estimates are the only ones reliably recovered from noisy data. Thus the $\hat{\beta}_0$ -image is a robustly smoothed version of the input. All the problems of LMedS estimators discussed in the previous section, however, are present and in the $\hat{\beta}_0$ -image discontinuities might be blurred.

Next, a nonoverlapping $N \times N$ ($N > 2n+1$) tessellation is defined. The tessellation delineates the windows in which an LMedS estimator is applied to the $\hat{\beta}_0$ -image. As initial regression coefficient estimates the set of N^2 values $\hat{\beta}_i, i = 1, \dots, p-1$ stored at the pixels in the window are used. Note that N^2 is of the same order of magnitude as the number of p -tuples recommended by the random sampling criterion (33). The estimates $\hat{\beta}_i, i = 0, \dots, p-1$ and $\hat{\sigma}$ are obtained. The former are the final robust estimates of the polynomial surface containing the majority of the pixels in the $N \times N$ window.

The estimate of the standard deviation $\hat{\sigma}$ is always an underestimate since it was derived from the smoothed $\hat{\beta}_0$ -image. The outliers in the $\hat{\beta}_0$ -image are outliers in the input image as well. The converse, however, is not true. Isolated impulses are removed by the $(2n+1) \times (2n+1)$ LMedS estimator. We detect the outliers of the $\hat{\beta}_0$ -image by computing the residuals relative to the final fit and using $\hat{\sigma}$ in (32). These outliers are marked on the original image. The residuals r_i of the input image relative to the estimated surface are also computed. The final standard deviation estimate $\hat{\sigma}$ is obtained from the input image by applying in the $N \times N$ window the following well known robust estimator to the pixels marked as inliers

$$\hat{\sigma} = 1.4826 \text{ med } |r_i - \text{med } r_i|. \quad (36)$$

The estimator (36) has a breakdown point of 0.5 and thus will tolerate the outliers not detected in the smoothed image.

For degree-0 surfaces the algorithm becomes simpler. The value of N should be taken sufficiently large that more than half the pixels in the window have $(2n+1) \times (2n+1)$ centered neighborhoods lying on only one surface. These pixels then carry reliable $\hat{\sigma}$ estimates. Recall that if a $(2n+1) \times (2n+1)$ window is centered on a step edge σ is strongly overestimated.

The zero order LMedS estimator is mode seeking, and thus both $\hat{\beta}_0$ and $\hat{\sigma}$ are obtained in the $N \times N$ window as the modes of the $\hat{\beta}_0$ - and $\hat{\sigma}$ -images.

The above condition imposes a lower bound on the value of N for a given n . For example, if $n = 2$ (5×5 window) we must have $N \geq 9$. The lower bound theoretically suffices only for edges aligned with the window (horizontal, vertical). Edges oriented along diagonals yield many $(2n+1) \times (2n+1)$ windows having only a few pixels belonging to the other surface. These windows may still provide good $\hat{\sigma}$ estimates but comparison of the final $\hat{\sigma}$ estimates from several windows should be used for corroboration. We return to the subject in the next section.

Two parameters of the algorithm can be used to indicate that the result of estimation within a $N \times N$ window is incorrect. Whenever either or both the following are violated, the returned estimates are not trustworthy:

- the number of inliers must be larger than half the window size;
- for noisy images $\hat{\sigma} \gg \tilde{\sigma}$.

This may happen when more than two surfaces are present in the processing window and no absolute majority for the pixels is available. The output of such windows should be discarded and the local image structure is recovered by region growing from the adjacent windows as will be shown in the next section.

In Table 4 the results of applying the degree-0 algorithm (both LMedS estimators are of degree-0) to a noisy vertical step edge are shown with $n = 2$ and $N = 9$. Two positions of the edge relative to the window were investigated. In the first case, only one column of the data is derived from the surface with amplitude 128. Thus the noiseless data has 9 outliers and $\epsilon = 0.11$. In the second case, the 9×9 window is centered on the edge. Five columns of pixels are from the surface with amplitude 168 and four columns from the surface with amplitude 128. The noiseless data has 36 outliers and $\epsilon = 0.44$. The step discontinuity was corrupted with Gaussian noise having $\sigma = 10$, and 100 trials were performed. Both the constant surface and the noise standard deviation are correctly estimated. Note the slight but significant improvement for the case of the 9×9 window centered over the edge. Since $h = 4\sigma$, a few noisy pixels belonging to the amplitude-128 surface may have values close to the amplitude-168 surface and vice versa. Thus the spread in the number of detected outliers is not necessarily due to incorrect decisions.

When the degree-1 algorithm is applied to the step discontinuity most of the 5×5 windows lying across it will return incorrect (tilted plane) estimates. If $\epsilon = 0.11$ the correct surface is still recovered because the majority of the pixels in the window belongs to the amplitude-168 surface. The average estimate values are similar to the ones obtained with the degree-0

Table 4: Results of the degree-0 algorithm applied to a step edge.

$\epsilon = 0.11$

Estimate	Min	Max	Mean	Std. dev.
$\hat{\beta}_0$	162.75	174.25	166.89	3.05
$\hat{\sigma}$	7.17	11.94	9.45	1.06
No. outliers	6	13	9.81	1.72

$\epsilon = 0.44$

Estimate	Min	Max	Mean	Std. dev.
$\hat{\beta}_0$	163.5	169.75	166.75	1.95
$\hat{\sigma}$	7.91	11.55	9.93	1.04
No. outliers	26	39	33	2.93

algorithm but their spread increases. If $\epsilon = 0.44$ the degree-1 algorithm fails because the number of undistorted pixels is no longer more than half the window size. A tilted plane is recovered from the data and only a very few outliers are detected in the smoothed image. The noise standard deviation is overestimated.

Table 5: Standard deviation estimates. Degree-1 algorithm applied to a step edge.

Case	Min	Max	Mean	Std. dev.
$\epsilon = 0.11$	7.07	13.37	10.1	1.62
$\epsilon = 0.44$	11.36	20.77	14.47	2.27

In Table 5 we compare the standard deviation estimates obtained with the degree-1 algorithm for the two edge positions. We can define the following σ -decision rule to determine the desirable estimator order for the analysis of the local image structure: compare the standard deviation estimates and take the results corresponding to the smaller value.

In our experiment when $\epsilon = 0.44$ the σ -decision rule chose the degree-0 estimator in all the 100 trials. The σ -decision rule is valid when the degree-1 results should be preferred. Significantly tilted planes or step and roof discontinuities between such planes all yield standard deviation overestimates when processed as degree-0 surfaces. The rule may not yield the degree-0 estimator for step edges at the borders of the window as can be seen by comparing Tables 4 and 5. However, in this case the fit recovered by the degree-1 algorithm is also close to the correct one. For horizontal surfaces the degree-1 algorithm is preferred since the local alignment of noisy data by a slightly tilted plane reduces the standard deviation. Again, the error is not significant and if the horizontal surface spans several adjacent windows, the fit is further improved after region growing.

In Figure 2d we showed the artifacts of LMedS estimators at corners. The proposed algorithm does not suffer from these artifacts. The image in the $N \times N$ window is redefined at the output as an inlier/outlier structure relative to the estimated fit. If the corner pixels are classified as outliers their fit can be assigned by region growing. When applied to the image in Figure 2a, the output of the algorithm is identical to the input. We have thus succeeded in extending the root-signal property discussed at the end of Section 3.2 to two dimensions.

4.3 Robust Region Growing

In this section we describe the application of the LMedS based image analysis technique to region growing. The region growing algorithm is only loosely connected with the local image structure recovery procedure discussed above and therefore it can also be used alone in applications involving fusion of similar patches.

In Section 4.2 we mentioned the warnings about possible estimation errors that the $N \times N$ window based LMedS estimator provides. If the noise homogeneously corrupts the entire image, an additional safeguard against errors can be secured. The mode of the $\hat{\sigma}$ values from the $N \times N$ windows is the robust (degree-0 LMedS) estimate of the corrupting noise standard deviation, say ω . As we have seen in Section 3.3 the standard deviation of *this* estimate is also returned by the estimator (31), say σ_ω . Then any window which yields $\hat{\sigma} > \omega + 2.5\sigma_\omega$ can be removed before region growing, i.e., its pixels are declared outliers. In these windows the large fitting error is probably due to an incorrect model. Recall that $\hat{\sigma}$ is already a robust estimate and the presence of a discontinuity in the window should not influence its value.

Our input in the region growing algorithm thus has the following characteristics. The image is tessellated with $N \times N$ windows. Inside a window a subset of pixels (always less than $N^2/2$) may have been declared as outliers, that is, they belong to a surface not represented by the fit recovered in the window. Windows in which warnings were generated are completely discarded and are regarded as containing only outliers. In our experiments (see Section 4.4) the number of such windows never exceeded 10 percent.

The region growing algorithm may run either on the original image or on the smoothed, $\hat{\beta}_0$ -image. In both cases, however, the same set of pixels are defined as outliers and the same operations are performed. The algorithm is iterative and has two steps per iteration.

Step 1. We can define 8-connected inlier and outlier components inside a window. In the first step of the region growing algorithm only the components containing at least one pixel at the window border present interest. Most of the pixels in such an outlier component belong to surfaces recovered in adjacent windows. (See Section 5 for a more careful analysis.) All the recovered surfaces are defined by a set of

parameters, and $\hat{\sigma}$ is the uncertainty about the fit. If the residual of an outlier pixel relative to the fit from an adjacent window is less than $2.5\hat{\sigma}$ the pixel may belong to that inlier component.

The expansion process of the inlier components, i.e., the definition of the outliers, must be performed in parallel. At every expansion step, the inlier components try to conquer a one pixel wide ring along their perimeter. Only the outlier pixels from adjacent windows are examined. To avoid biasing the fit, the parameters of the initial surface are not updated during the expansion. The expansion process stops when all the pixels of the outlier components have been conquered, or when no more expansion is possible. If more than one expanding inlier component reaches a pixel simultaneously, the component yielding the smallest residual/ $\hat{\sigma}$ ratio wins. The expansion process cannot take more than N steps.

The fit and standard deviation estimates are recomputed for the new, expanded regions. Since these regions should not contain discontinuities, a simple least squares applied to the inliers (outliers could exist inside the region due to impulse noise) suffices. The use of LMedS estimator may, however, provide warnings about an erroneous expansion process. Several outliers, not claimed by any neighboring inlier component, can remain, especially in noisy images. The initial influence of the window tessellation on the result of the expansion process is removed by later iterations of the algorithm.

Step 2. At the beginning of the second step of the region growing algorithm, most of the image is tessellated by regions of inliers. Our goal is to fuse in parallel the best matching pairs of contiguous regions. Let $A_0, \hat{\sigma}_0$ be the area (number of inlier pixels contained within) and the standard deviation of an inlier region. Assume that the region has common borders with $j=1, 2, \dots, m$ other inlier regions. Let $A_j, \hat{\sigma}_j$ be the corresponding parameters of such a region.

For the concatenated region obtained from the pair $0, j$ we re-estimate the underlying image structure. This can be achieved by either applying the full algorithm described in Section 4.2 or only the degree-0 or degree-1 procedures. Robust techniques must be employed since the two regions may belong to different surfaces. Let $\hat{\sigma}_{0,j}$ be the robust standard deviation estimate of the concatenated $(0, j)$ region. The new inlier/outlier structure is then defined within both the 0 and the j region. Let $A_{0,j}$ be the area of the new inlier connected component in region 0 , and $A_{j,0}$ be the area of the new inlier connected component in region j . These new inlier components cannot be larger than the original regions; thus

$$A_{0,j} \leq A_0 \quad A_{j,0} \leq A_j. \quad (37)$$

Note that the sum of $A_{0,j}$ and $A_{j,0}$ gives the area of the inlier component in the concatenated region $(0, j)$. The efficacy of region growing can be measured either by the increase in the area or by the reduction of the standard deviation, of the concatenated region relative to its parts. We define

$$\eta_A(0, j) = \min \left[\frac{A_{0,j}}{A_0}, \frac{A_{j,0}}{A_j} \right] \quad (38)$$

$$\eta_o(0, j) = \max [T_o, \min (\hat{\sigma}_0, \hat{\sigma}_j)]$$

where T_o is an absolute threshold of smoothness. The $0, j$ regions are considered as candidates for fusion if

$$\eta_A(0, j) > 0.8 \quad (39)$$

or

$$\eta_A(0, j) > 0.5 \text{ and } \hat{\sigma}_{0,j} < \eta_o(0, j). \quad (40)$$

The condition (39) enables fusion when the regions extend one over the other almost completely, i.e., the represented surfaces are very similar. There is no need for a condition on smoothness. However, if the newly obtained region of inliers has an area which is just larger than the average of the areas of the initial regions, a smoothness criterion must also be satisfied (40). The new standard deviation should not exceed the standard deviations of the parts or an absolute smoothness threshold.

Definitions (38) are symmetrical and a processing scheme sequential at the paired regions level but parallel over the image must be designed. The adjacency relations among the inlier regions can be described by an undirected graph and the maximal independent set of this graph is the sought solution. We did develop a parallel algorithm which solves the problem in a few steps (Meer 1989, Montanvert *et al.* 1989) but its description is beyond the goal of this paper. From all the candidates the neighbor indexed by

$$k = \arg \min_j \hat{\sigma}_{0,j} \quad (41)$$

is chosen for fusion with region 0. Criterion (41), however, is not symmetrical. The region k can choose as its best fusion candidate a neighbor not contiguous to region 0. Thus, "chains" of regions trying to fuse may be obtained. The "chains" are described by directed graphs with the arcs weighted by $\hat{\sigma}_{0,j}$. Since these graphs are a few arcs long, we allowed only the fusion of the pair having the smallest weight along the path. Using the already mentioned technique (Meer 1989), a parallel solution in which more than one pair is fused can be obtained. Further iterations of the region growing algorithm, however, makes the effort superfluous.

At the end of the second region growing step the image contains larger inlier components and some new outlier components with pixels rejected during fusion. These outliers will be defined at the first step of the next iteration, as was shown above. Note that the artifacts created by the initial $N \times N$ tessellation of the image are reduced with each iteration, as the robust image analysis shifts the region boundaries toward the discontinuities in the image. With parallel fusion procedures a large homogeneous surface is recovered in a logarithmic number of steps. Indeed, at every iteration two adjacent regions are fused, yielding exponential growth.

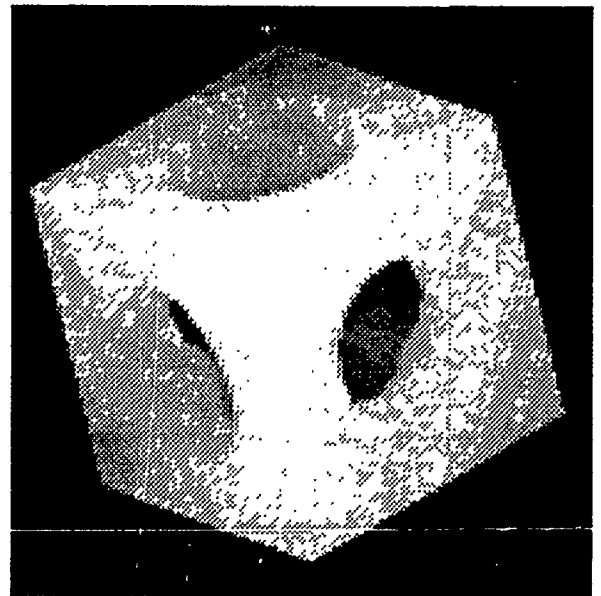
Region growing stops when no more pairs of inlier

components can be fused and outliers conquered. In our experiments the number of iterations never exceeded ten. Some of these experiments are described in the next section.

4.4 Experimental Results

In the experiments the local image structure analysis (Section 4.2.) employed a 5×5 window at every pixel and a 9×9 window based tessellation of the image. Region growing (Section 4.3.) was performed on the smoothed β_0 -image and the parameters were estimated by the degree-1 algorithm. The absolute smoothness threshold was $T_o = 0.4$ yielding a minimum residual for an outlier pixel of one gray level.

In Figure 4a the synthetic range image of a cube with three holes is shown. The image is noiseless; we use it only to illustrate the evolution of the region growing. The tessellation of the image after the first iteration of the region growing algorithms is shown in Figure 4b. That is, the local inlier/outlier structures were determined, most of the outliers were defined by adjacent windows, and some small regions were already fused. The original 9×9 tessellation can be recognized in Figure 4b. Significant edges of the cube, however, are already delineated as region boundaries. After five iterations the tessellation shown in Figure 4c is obtained. The faces of the cube and the background are correctly delineated. The result after the last (ninth) iteration is shown superposed on the input image in Figure 4d. As was mentioned in Section 4.3. the fusion was not implemented in a completely parallel procedure, and several iterations are necessary to fuse the few regions forming the background. The non-planar surfaces (inside the holes) are segmented

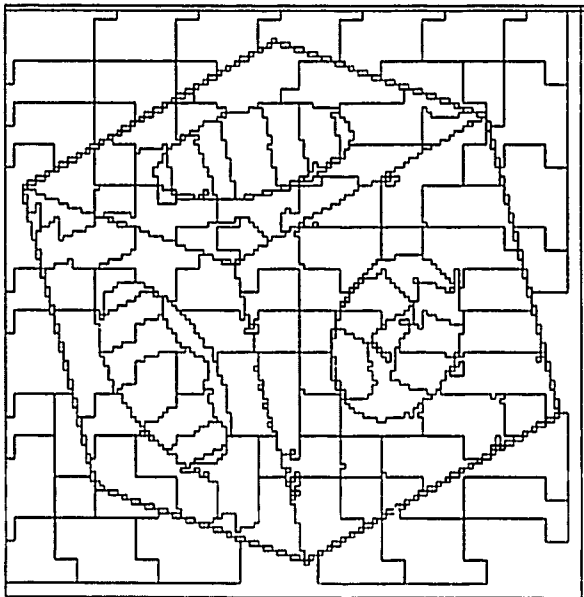


a) Original.

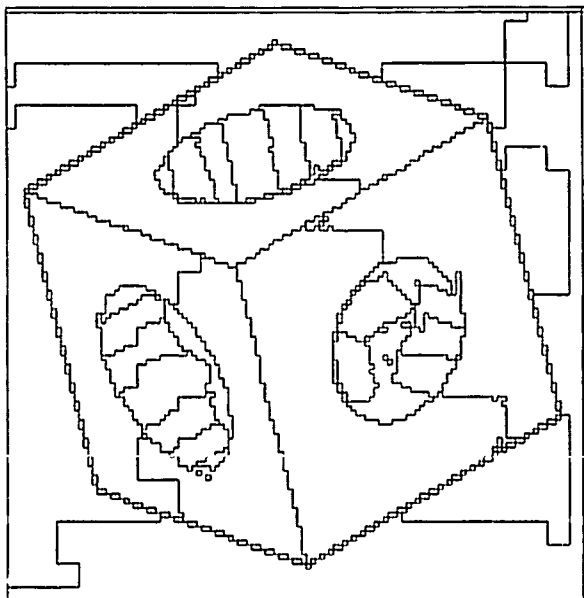
Figure 4: Segmentation of a synthetic range image of a cube with three holes.

somewhat arbitrarily by the planar model. Many of these pixels, however, are retained as outliers (Figure 4e) underlining the robustness of the region growing. In this picture the inlier pixels take their values from the recovered fits.

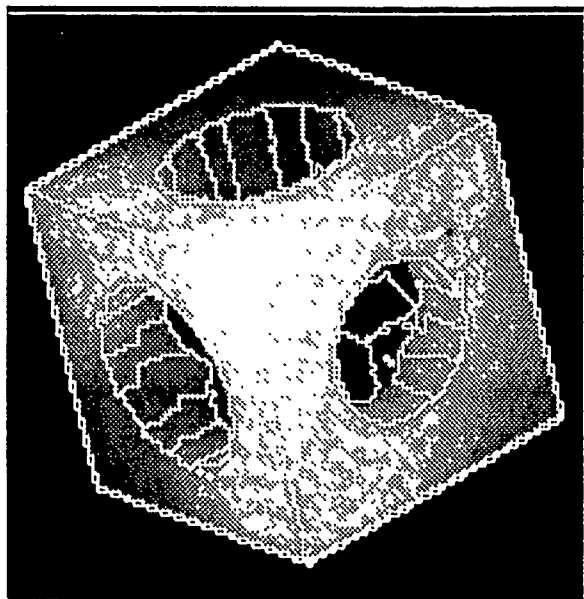
In Figure 5a the same range image is shown corrupted with Gaussian noise having $\sigma = 6$. This noise level yields a very low signal-to-noise ratio at the roof edges of the image (Section 3.1.). The smoothed β_0 -image on which the region growing was performed is shown in Figure 5b. In Figure 5c the final segmentation (after seven iterations) is given. Pixels retained as outliers are shown in Figure 5d together with the



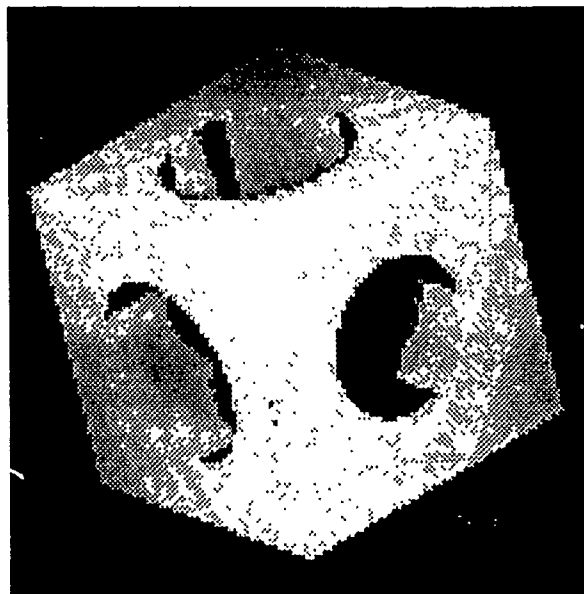
b) Tessellation after one iteration of the region growing algorithm.



c) Same after five iterations.



d) Final segmentation superposed on the input.

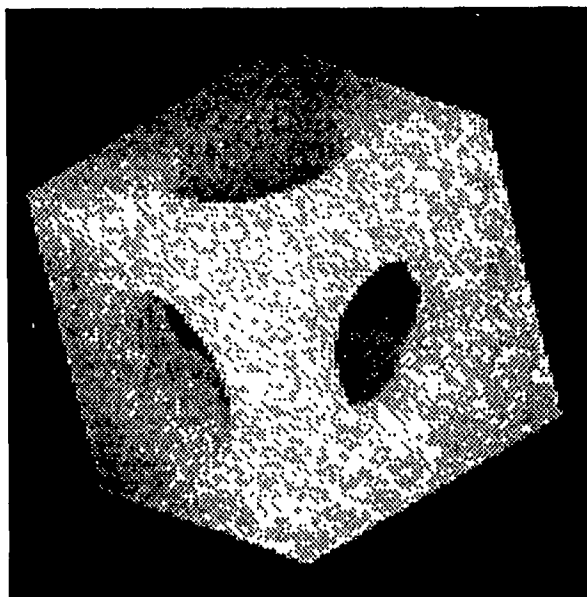


e) Reconstructed input.

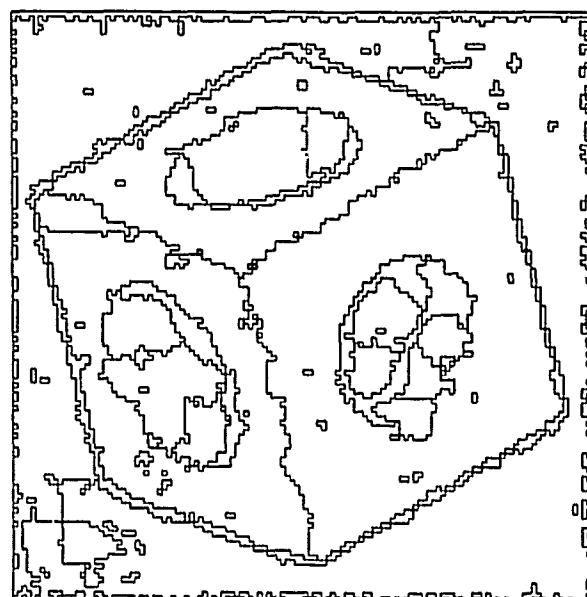
Outlier pixels marked with black on the cube. They are not classified as background.

recovered fits. Note that the background is not recovered with zero gray level since the clipping of the Gaussian noise yields non-zero mean. A comparison with the original image (Figure 4a) helps to assess the quality of the reconstruction.

The range image of a ring on steps obtained with an ERIM sensor is shown in Figure 6a and its smoothed version in Figure 6b. After nine iterations, the boundaries of the recovered regions, superposed on the smoothed image, are shown in Figure 6c. Note that the upper surface of the ring is delineated as one region. Several pixels on non-planar surfaces or at the noisy roof edges were declared outliers (Figure 6d). The reconstruction of the input is also given in

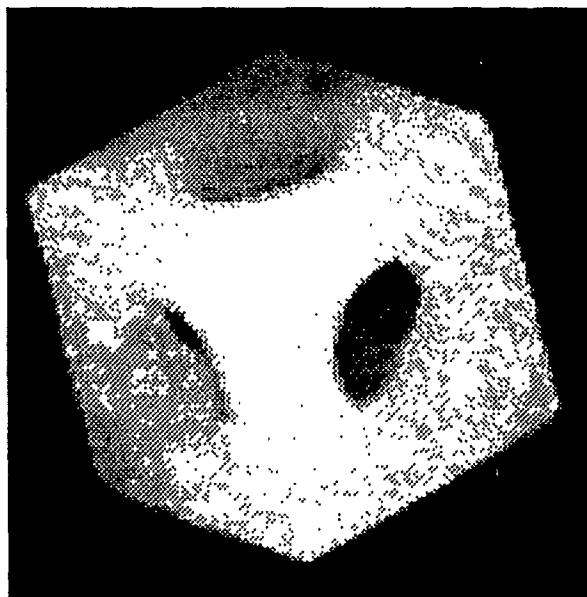


a) Input.

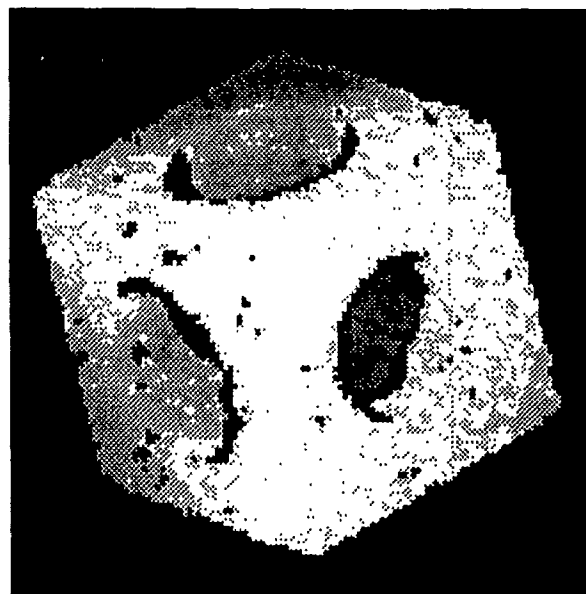


c) Delineated regions after seven iterations.

Figure 5: Segmentation of the noisy cube image ($\sigma = 6$).



b) Smoothed image.



d) Reconstructed input.

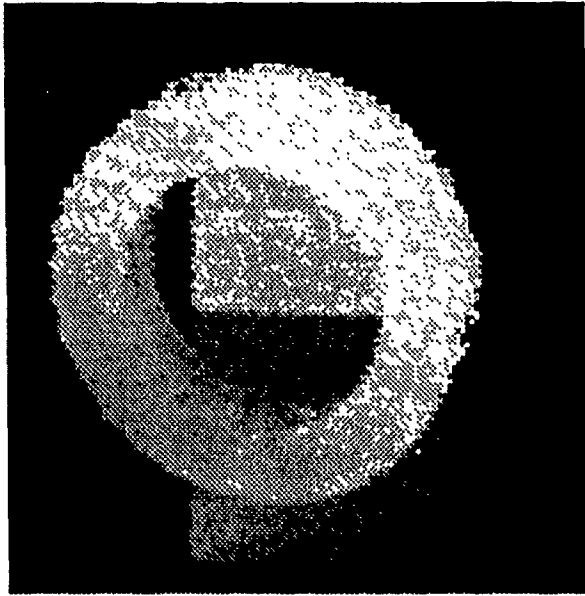
(Pixels retained as outliers are marked with black.)

Figure 6d.

All the previous examples used range images. The frequent roof edges make range images difficult for traditional image analysis techniques. Our segmentation results are more fragmented than the outputs from similar images obtained by Besl and Jain (1988) with a different method. (The cube is noisier, the ring image is the same.) We did not use any postprocessing; our goal is to use image structure analysis as an example for the potential of the LMedS paradigm. Also, our surface model was restricted to planar fits, while Besl and Jain used up to biquartic surfaces. In noisy images, higher-order surface fits achieve less smooth-

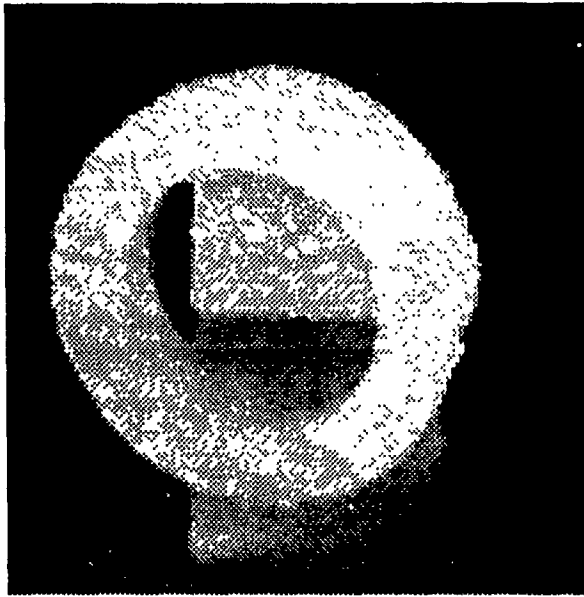
ing. The robust technique allows us to maximize the amount of smoothing by employing only planar fits for image analysis. Non-planar surfaces are then broken along natural boundaries; see for example the inner parts of the holes in the cube (Figures 4d and 5c). A robust postprocessing procedure applied to regions where the local structure suggests the presence of a higher-order surface can then be used. The definition of the optimum robust model order is an open problem in statistics, although several methods for doing it have been developed (Hampel *et al.*, 1986, p. 366).

In Figure 7a the gray level image of a house is shown. The obtained segmentation, superposed on the input (Figure 7b), correctly identifies the long elongated features. These features are defined as



a) Input.

Figure 6: Segmentation of a range image showing a ring on steps.



b) Smoothed image.

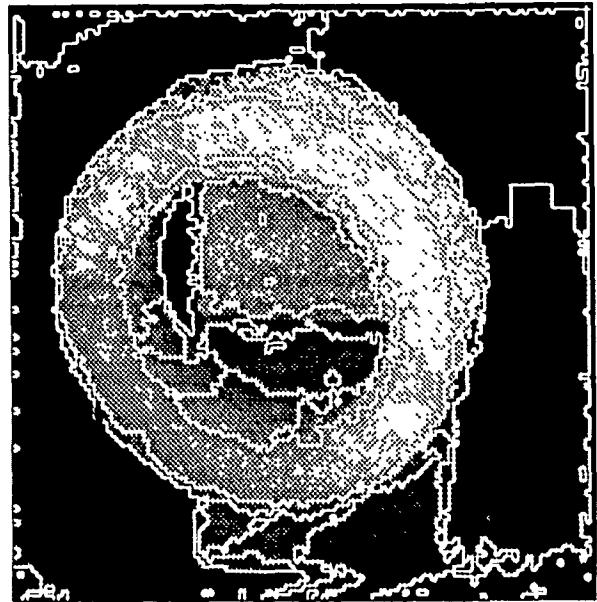
outliers at the output. The performance of the algorithm can be assessed from the output of the aerial image (Figure 8). In this case the degree-0 algorithm was used for estimation. Note that all the large and/or high contrast features are correctly delineated.

Robust image analysis is computationally expensive if not implemented on a parallel machine. In spite of the Monte Carlo speed-up (Section 3.3) it takes about 40 minutes of CPU time to obtain the initial estimates $\hat{\beta}_i, \hat{\sigma}$ at every pixel of a 128×128 image. The same procedure, however, requires less than one minute on the Connection Machine where pixel level parallelism can be achieved. Parallel, hierarchical implementation

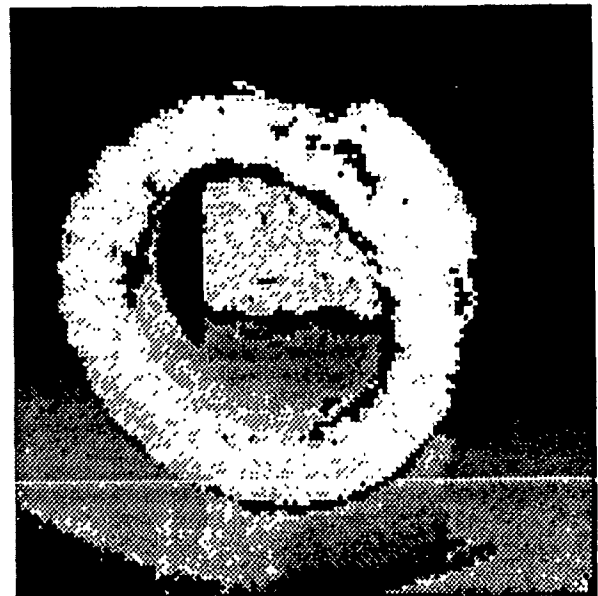
of the later, non-local stages of the algorithm (region growing) is possible using the data driven pyramids techniques we developed in the context of a different problem (Montanvert *et al.* 1989). The simulated parallel implementation on a VAX 785 requires less than six seconds on the average for the definition of a pair of regions to be fused (analysis of all the neighbors).

5 Discussion

In this paper we have presented the mathematical background of Least Median of Squares (LMedS) esti-



c) Final segmentation superposed on the smoothed image.



d) Reconstructed input. Pixels retained as outliers are marked with black.



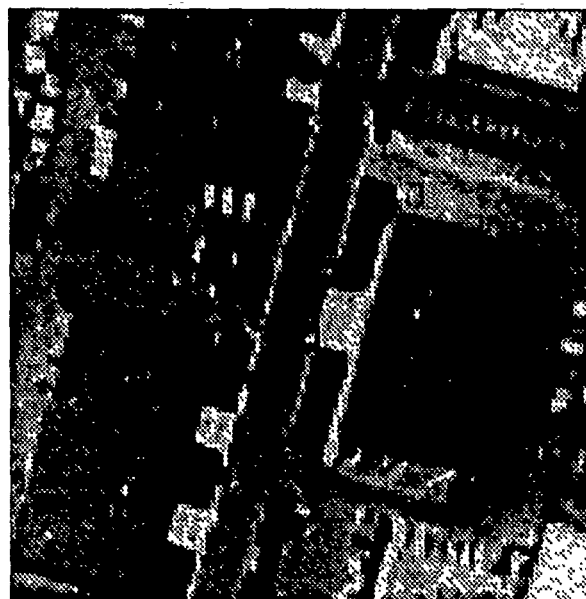
a) Original.



b) Segmentation superposed on the input.

Figure 7: Segmentation of the house gray level image. The minimization criterion (18) separates the data into two classes: inliers on which the recovered surface fit is based, and outliers whose origin cannot be determined. Thus, the LMedS estimators will return perfect estimates whenever the inlier data is not corrupted by noise. For example, the use of a LMedS based technique in template pattern matching should eliminate the artifacts of least squares type cost functions.

We used the problem of image structure recovery as example for developing a LMedS based algorithm. Since the clear dichotomy between "good" and "bad" data does not exist in noisy images, the problem is worst-case for the high breakdown point robust



a) Original.



b) Segmentation superposed on the input.

Figure 8: Segmentation of the aerial gray level image. The local-to-global image recovery technique described above is not necessarily optimal. Same quality segmentations can be obtained by a dual, global-to-local technique involving clustering in feature space. We developed such a technique, again using LMedS estimators (Jolion *et al.* 1990).

The images in this paper are scalar-valued and defined on a regular square lattice. This is not a prerequisite for the application of LMedS estimators. Sparsely defined data points and/or vector-valued pixels can be processed by the same method. We performed preliminary experiments applying LMedS estimators to stereo disparity detection and texture segmentation (sparse data), as well as optical flow

decomposition (sparse vector-valued data).

The LMedS estimator is a good candidate for an edge detection operator. When applied to an edge, the high breakdown point robust operator selects the surface to which the majority of pixels belong. Note that no differentiation process is employed in the recovery of the inliers' fit. Depending on the size of the operator's support the same inlier/outlier structure is recovered in several shifted positions around the edge. Thus, the operator can be applied at every pixel and the inlier/outlier descriptions of the same region stored pixelwise in an accumulator. By locating the border between the inlier and outlier connected components in the accumulator detection of edges in noisy images is achieved. The method is not restricted to step edges since the operator can employ higher order models as well.

Features smaller than the half the window size employed in the analysis (9×9 in our examples) are discriminated as outliers. If these features are well defined (i.e., have high contrast) they remain delineated as outliers during region growing. Thus, by tracing an outlier connected component, features can be recovered below the expected resolution of the analysis. We classified a pixel as an inlier or outlier by a binary decision (32). The value of the normalized residual can be used to define an outlierness measure. Description of the local image structure by this continuous measure may be of interest in certain applications.

In conclusion, we believe that the LMedS estimator is a valuable tool for solving computer vision problems. Its application, however, should be preceded by a careful analysis of the characteristics of the available data. If the two classes (signal and noise) are clearly separable the LMedS estimator yields superior results.

Acknowledgement

We are grateful to Dong Yoon Kim for discussions in earlier stages of the research and to Paul Besl for providing us with most of the images used in the examples.

References

- [Aho *et al.*, 1974] A.V. Aho, J.E. Hopcroft and J.D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA.
- [Bajcsy *et al.*, 1986] R. Bajcsy, M. Mintz and E. Lieberman. A common framework for edge detection and region growing. MS-CIS-86-13, Dept. of Computer and Information Science, University of Pennsylvania, Philadelphia.
- [Basseville *et al.*, 1981] M. Basseville, B. Espiau and J. Gasnier. Edge detection using sequential methods for change in level—Part I: A sequential edge detection algorithm. *IEEE Trans. Acoustics, Speech, Signal Processing ASSP-29*, 24–31.
- [Basseville and Benveniste, 1986] M. Basseville and A. Benveniste, eds. *Detection of Abrupt Changes in Signals and Dynamical Systems*. Springer, NY.
- [Bergholm, 1987] F. Bergholm. Edge focusing. *IEEE Trans. Pattern Analysis Machine Intelligence PAMI-9*, 726–741.
- [Besl and Jain, 1988] P.J. Besl and R.C. Jain. Segmentation through variable-order surface fitting. *IEEE Trans. Pattern Analysis Machine Intelligence PAMI-10*, 167–192.
- [Besl *et al.*, 1988] P.J. Besl, J.B. Birch and L.T. Watson. Robust window operators. *Proceedings of the Second International Conference on Computer Vision*, December 5–8, 1988, Tampa, Florida, 591–600. See also *Machine Vision and Applications* 2, 1989, 179–214.
- [Blake, 1989] A. Blake. Comparison of the efficiency of deterministic and stochastic algorithms for visual reconstruction. *IEEE Trans. Pattern Analysis Machine Intelligence PAMI-11*, 2–12.
- [Canny, 1986] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis Machine Intelligence PAMI-8*, 679–698.
- [Cavanagh and Leclerc, 1989] P. Cavanagh and Y.G. Leclerc. Shape from shadows. *Journal of Experimental Psychology: Human Perception and Performance* 15, 3–27.
- [Chakraborti and Misra, 1987] N.B. Chakraborti and R. Misra. Transition detection in image processing. *Signal Processing* 13, 197–207.
- [Chen *et al.*, 1989] M.H. Chen, D. Lee and T. Pavlidis. Residual analysis for edge detection. TR.89.05.15.R, Department of Computer Science, SUNY at Stony Brook.
- [Chow, 1960] G.C. Chow. Tests of equality between sets of coefficients in two linear regressions. *Econometrica* 28, 591–605.
- [Cornsweet, 1970] T. Cornsweet. *Visual Perception*. Academic Press, NY.
- [Coyle *et al.*, 1989] E.J. Coyle, J.H. Lin and M. Gabbouj. Optimal stack filtering and the estimation and structural approaches to image processing. *IEEE Trans. Acoustics, Speech, Signal Processing ASSP-37*, 2037–2066.
- [Dattatreya and Kanal, 1988] G.R. Dattatreya and L.N. Kanal. Detection and smoothing of edge contours in images by one dimensional Kalman filtering techniques. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 11–15, 1988, New York; 1052–1055.
- [Edelsbrunner and Souvaine, 1988] H. Edelsbrunner and D.L. Souvaine. Computing median-of-squares regression lines and guided topological sweep.

- Report UIUCDCS-R-88-1483, Department of Computer Science, University of Illinois at Urbana-Champaign.
- [Fischler and Bolles, 1981] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM* **24**, 381-395.
- [Gelb, 1974] A. Gelb, ed. *Applied Optimal Estimation*. M.I.T. Press, Cambridge, MA.
- [Hampel et al., 1986] F.R. Hampel, E.M. Ronchetti, P.J. Rousseeuw and W.A. Stahel. *Robust Statistics: An Approach Based on Influence Functions*. John Wiley & Sons, NY.
- [Haralick and Joo, 1988] R.M. Haralick and H. Joo. 2D-3D pose estimation. *Proceedings of the 9th International Conference on Pattern Recognition*, November 14-17, 1988, Rome, Italy, 385-391. See also *IEEE Trans. Systems, Man, Cybernetics SMC-19*, 1426-1446, 1989.
- [Haralick and Watson, 1981] R.M. Haralick and L. Watson. A facet model for image data. *Computer Graphics Image Processing* **15**, 113-129.
- [Haralick and Shapiro, 1985] R.M. Haralick and L.G. Shapiro. Image segmentation techniques. *Computer Vision Graphics Image Processing* **29**, 100-132.
- [Haralick et al., 1983] R.M. Haralick, L.T. Watson and T.J. Laffey. The topographic primal sketch. *Int. J. Robotics Research* **2**, 50-72.
- [Harwood et al., 1987] D. Harwood, M. Subbarao, H. Hakalahti and L.S. Davis. A new class of edge-preserving smoothing filters. *Pattern Recognition Letters* **6**, 155-162.
- [Heinonen and Neuvo, 1988] P. Heinonen and Y. Neuvo. FIR-Median hybrid filters with predictive FIR substructures. *IEEE Trans. Acoustics, Speech, Signal Processing ASSP-36*, 892-899.
- [Huber, 1981] P.J. Huber. *Robust Statistics*. Wiley, NY.
- [Jolion et al., 1990] J.M. Jolion, P. Meer and A. Rosenfeld. Generalized minimum volume ellipsoid clustering with applications in computer vision. Computer Vision Laboratory, University of Maryland, College Park, CAR-TR-485.
- [Kashyap and Eom, 1988] R.L. Kashyap and Kie-Bum Eom. Robust image models and their applications. In *Advances in Electronics and Electron Physics*, Vol. 70, Academic Press, San Diego, CA, 80-157.
- [Kim et al., 1989] D.Y. Kim, J.J. Kim, P. Meer, D. Mintz and A. Rosenfeld. Robust computer vision: A least median of squares based approach. In *Proceedings of the DARPA Image Understanding Workshop*, May 23-26, 1989, Palo Alto, CA, 1117-1134.
- [Koenderink and van Doorn, 1982] J.J. Koenderink and A.J. van Doorn. The shape of smooth objects and the way contours end. *Perception* **11**, 129-137.
- [Koivo and Kim, 1989] A.J. Koivo and C.W. Kim. Robust image modeling for classification of surface defects on wood boards. *IEEE Trans. Systems, Man, Cybernetics SMC-19*, 1659-1666.
- [Kumar and Hanson, 1989] R. Kumar and A.R. Hanson. Robust estimation of camera location and orientation from noisy data having outliers. *Proceedings of the Workshop on Interpretation of 3D Scenes*, November 27-29, 1989, Austin, TX, 52-60.
- [Kunt et al., 1987] M. Kunt, M. Bènard and R. Leonardi. Recent results in high-compression image coding. *IEEE Trans. Circuits Systems CAS-34*, 1306-1336.
- [Leclerc, 1988] Y.G. Leclerc. Constructing simple stable descriptions for image partitioning. In *Proceedings of the DARPA Image Understanding Workshop*, April 6-8, 1988, Cambridge, MA, 365-382.
- [Leclerc, 1989] Y.G. Leclerc. Image and boundary segmentation via minimal-length encoding on the Connection Machine. In *Proceedings of the DARPA Image Understanding Workshop*, May 23-26, 1989, Palo Alto, CA, 1056-1069.
- [Leclerc and Zucker, 1987] Y.G. Leclerc and S.W. Zucker. The local structure of image discontinuities in one dimension. *IEEE Trans. Pattern Analysis Machine Intelligence PAMI-9*, 341-355.
- [Lee et al., 1989] C.N. Lee, R.M. Haralick and X. Zhuang. Recovering 3-D motion parameters from image sequences with gross errors. *Proceedings of the Workshop on Visual Motion*, March 20-22, 1989, Irvine, CA, 46-53.
- [Lee, 1988] D. Lee. Coping with discontinuities in computer vision: their detection, classification and measurement In *Proceedings of the Second International Conference on Computer Vision*, December 5-8, 1988, Tampa, Florida, 546-557.
- [Lee, 1989] D. Lee. Edge detection, classification and measurement. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 4-8, 1989, San Diego, CA, 2-10.
- [Lee et al., 1988] D. Lee, T. Pavlidis and K. Huang. Edge detection through residual analysis. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 5-9, 1988, Ann Arbor, MI, 215-222.

- [Meer, 1989] P. Meer. Stochastic image pyramids. *Computer Vision, Graphics, and Image Processing*, 45, 269-294.
- [Meer *et al.*, 1990] P. Meer, D. Mintz, D.Y. Kim and A. Rosenfeld. Robust regression in computer vision: Least median of squares estimators. Submitted to *International Journal of Computer Vision*.
- [Mintz and Amir, 1989] D. Mintz and A. Amir. Generalized random sampling. CAR-TR-441, Computer Vision Laboratory, University of Maryland, College Park.
- [Montanvert *et al.*, 1989] A. Montanvert, P. Meer, A. Rosenfeld. Hierarchical image analysis using irregular tessellations. CAR-TR-464, Computer Vision Laboratory, University of Maryland, College Park.
- [Mowforth *et al.*, 1987] P.H. Mowforth, J. Jelinek and Z.P. Jin. An appropriate representation for early vision. *Pattern Recognition Letters* 5, 175-182.
- [Nalwa and Binford, 1986] V.S. Nalwa and T.O. Binford. On detecting edges. *IEEE Trans. Pattern Analysis Machine Intelligence*, PAMI-8, 699-714.
- [Pavlidis and Lee, 1989] T. Pavlidis and D. Lee. Residual analysis for feature extraction. In *From Pixels to Features*, J.C. Simon, ed. North-Holland, Amsterdam, 219-227.
- [Pavlidis and Liow, 1988] T. Pavlidis and Y.T. Liow. Integrating region growing and edge detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 5-9, 1988, Ann Arbor, MI, 208-214. See also *IEEE Trans. Pattern Analysis Machine Intelligence*, PAMI-12, 225-233, 1990.
- [Pednault, 1989] E.P.D. Pednault. Some experiments in applying inductive inference principles to surface reconstruction. In *Proceedings of the International Joint Conference on Artificial Intelligence*, August 20-25, Detroit, MI, 1603-1609.
- [Perona and Malik, 1987] P. Perona and J. Malik. Scale space and edge detection using anisotropic diffusion. In *Proceedings of the IEEE Computer Society Workshop on Computer Vision*, November 30-December 2, 1987, Miami, FL, 16-22. See also Report UCB/CSD 88/483, University of California, Berkeley.
- [Rousseeuw, 1984] P.J. Rousseeuw. Least median of squares regression. *J. Amer. Stat. Assoc.* 79, 871-880.
- [Rousseeuw and Leroy, 1987] P.J. Rousseeuw and A.M. Leroy. *Robust Regression & Outlier Detection*. Wiley, NY.
- [Saint-Marc and Medioni, 1988] P. Saint-Marc and G. Medioni. Adaptive smoothing for feature extraction. In *Proceedings of the DARPA Image Understanding Workshop*, April 6-8, 1988, Cambridge, Mass., 1100-1113.
- [Sinha and Schunck, 1989] S.S. Sinha and B.G. Schunck. A two stage algorithm for discontinuity-preserving surface reconstruction. Artificial Intelligence Laboratory, University of Michigan, Ann Arbor, preprint.
- [Tirumalai and Schunck, 1988] A. Tirumalai and B.G. Schunck. Robust surface approximation using least median squares regression. CSE-TR-13-89, Artificial Intelligence Laboratory, University of Michigan, Ann Arbor.
- [Wilks, 1962] S.S. Wilks. *Mathematical Statistics*, Wiley, NY.
- [Yokoya and Levine, 1989] N. Yokoya and M.D. Levine. Range image segmentation based on differential geometry: A hybrid approach. *IEEE Trans. Pattern Analysis Machine Intelligence* PAMI-11, 643-649.

A FAST, HIGH BREAKDOWN POINT ROBUST ESTIMATOR FOR COMPUTER VISION APPLICATIONS

Doron Mintz, Peter Meer and Azriel Rosenfeld

Computer Vision Laboratory, Center for Automation Research,
University of Maryland, College Park, MD 20742-3411

ABSTRACT

This short note presents a fast algorithm for the computation of parametric models in computer vision applications; it supplements the preceding paper [2] in these Proceedings. The algorithm is similar to the Least Median of Squares (LMedS) estimator (which it uses extensively as a computation module) and preserves its high breakdown point property. Reduced computational time is achieved by recognizing that in many computer vision problems the data is defined on a square sampling lattice. In such a case, the model estimation can be decomposed into independent degree-0 LMedS estimation problems. Since degree-0 LMedS is much faster than its higher degree counterparts, the new estimator yields a significant speedup without perceptible degradation in the quality of the output. The algorithm should make it feasible to use high breakdown point robust methods in on-line computer vision systems.

1. Introduction

The Least Median of Squares (LMedS) estimator has become increasingly popular in computer vision [3], primarily due to its high breakdown point, which is close to the theoretical upper bound. LMedS can in principle recover the parametric model (surface) representing half of the data in the processing window, even when the rest of the window is severely corrupted by noise (outliers). However, we have shown [2] that the presence of (say) Gaussian noise corrupting all the pixels increases the sensitivity of LMedS to outliers. Also, transitions in images are not always sharp, and at such transitions often none of the models accounts for half of the data in the window.

In [2] we described a two-stage algorithm that overcomes most of the difficulties with LMedS. It uses a random sampling technique to reduce the

computational cost of LMedS, but it is still quite slow and its usefulness for on-line applications is limited. However, if we take into account the special structure of most computer vision problems, a fast algorithm that preserves the high breakdown point of LMedS can be developed.

There is a strong similarity between the computation of LMedS estimators and the RANSAC paradigm [1], as pointed out in [3]. The proposed new fast algorithm may therefore be useful for speeding up applications employing RANSAC as well.

2. Motivation

In most computer vision problems the parametric model to be estimated is a function of the image coordinates. Note that these variables are independent, and that the denseness or sparseness of the data is not relevant to our discussion. The data is assumed to be linear in the model parameters and additively corrupted by white noise (Eq. 2 in [2]). This model includes polynomial surface fitting, since the regressor variables (Eq. 1 in [2]) can be monomials in the coordinates. It is always possible to find an orthogonal base for the vector space of the processing window [4], and to define the model relative to this base. When the resulting model parameters are estimated by least squares regression they are unbiased and uncorrelated, i.e., their ensemble mean is correct and their covariance matrix is diagonal [5]. See also the example in [2], Section 3.1.

In the original LMedS method the data is projected onto the subspace of one of the desired parameters and a cost function (the median of the squared residuals) is computed. This procedure is repeated several times with the goal of minimizing the cost function. In our new approach, most of the parameter estimates are obtained independently by successive application of degree-0 LMedS estimators.

3. Algorithm

We consider every data point in the processing window as the center of a small neighborhood. There should be enough points in this neighborhood to allow least squares estimation of the local model parameters. For example, a 3×3 neighborhood suffices to recover the three parameters of a plane from the data on a dense grid. The neighborhood estimates can be stored in separate *parameter windows* defined on the same sampling lattice. The estimates are uncorrelated and therefore the parameter windows can be processed independently.

At this stage of our research we have restricted ourselves to piecewise planar models. In this situation the locally estimated slopes of the planes are noisy constants, as long as the neighborhood does not include data from another model. Therefore, after the local least squares estimation the slope windows are either constant or consist of a blurred step edge corrupted by noise. Under the same conditions the intercepts of the planes vary linearly with the coordinates. Note that the three variables of the planar model themselves form an orthogonal base. In the general polynomial case, only the highest degree coefficients remain constant when estimated in small neighborhoods over a homogeneous region.

Consider a parameter window containing a blurred step edge. Away from the edge the neighborhoods have the correct models. In our algorithm these neighborhoods are selected from the window without explicitly delineating the position of the discontinuity. The variance of the residual relative to the local mean is estimated in the same neighborhoods. Under the assumption of stationary noise in the window, all the neighborhoods away from the edge yield variance values from the same random process. Therefore the variance distribution over the window is approximately unimodal, and a degree-0 LMedS estimator (i.e., mode seeking) is "well behaved" [2]. The estimator returns a reliable variance value for the parameter window, and also separates the data into points which supplied reliable variance estimates and points which did not. The former points are selected as reliable for estimating the parameter of that parameter window.

The selected data comes only from the distinct models; i.e., the regions of smooth transition between models have been eliminated. The clear dichotomy facilitates degree-0 LMedS estimation, and the parameter estimate corresponding to the majority of the *selected* points in the parameter window is obtained. Its value is the parameter estimate used for the model. The points in the parameter window belonging to the model, i.e., supplying the estimate, are marked as the inliers of that parameter.

In the current (piecewise planar) implementation of the algorithm only the intercept cannot be estimated with the above described procedure. As in the LMedS method the data is projected onto the intercept

subspace using the already available model parameter values. Then the projected points which correspond to inliers in both slope windows are selected. The value of the intercept is obtained by again applying a degree-0 LMedS estimator. The problem of higher order polynomial fits is currently under study.

The speedup of the algorithm is due to the elimination of the random sampling in LMedS [2]. Since, at least for a piecewise planar structure, all the parameters are found by mode seeking, the complexity is reduced to the amount of computation previously required for one sample. Furthermore, the piecewise constant parameter windows can be processed in parallel.

The following is a summary of the piecewise planar algorithm for dense data.

1. Define an $N \times N$ processing window.
2. At every pixel obtain by a least squares procedure in a centered $(2n+1) \times (2n+1)$ neighborhood the two slope estimates of the local plane. This yields two slope windows.
3. For each slope window:
 - 3.1. Estimate the variance of the residuals relative to the mean in the $(2n+1) \times (2n+1)$ neighborhood.
 - 3.2. Estimate the mode of the variance distribution by a degree-0 LMedS estimator. Select the pixels within a given range from the mode.
 - 3.3. Estimate the slope parameter of the window from the selected pixels by a degree-0 LMedS estimator.
4. Project the data in the window onto the subspace of the intercept. Select the points corresponding to inliers in both slope images.
5. Repeat 3.3 for the projected data.
6. (Optional) The complete model can serve as an initial estimate for a reweighted least squares or an M-estimator based refinement to increase the efficiency of the method.

4. Experiments

The superior performance of the new algorithm is due to the variance-based data selection process in its computational module (step 3 above). In Table 1 a comparison between the output of this module and the original degree-0 LMedS estimator is given. The 9×9 processing window contained a step edge (gray level values 128 and 178), there were four columns of 128 and five columns of 178 ($\epsilon = 0.44$). The edge was corrupted with zero mean Gaussian noise with standard

deviation σ . The mode and the spread (standard deviation of the mode) of the distribution of estimates in 100 trials were recorded. The increased robustness of the estimates computed with the new method can be seen in Table 1.

For comparison with the results in [2], the new algorithm was implemented using 3×3 neighborhoods defined in 9×9 windows. (The latter size can be reduced.) All the images in [2] were processed. On a 128×128 image the previous method required at least 40 minutes of CPU time on a VAX 785. A non-optimized implementation of the new algorithm on the VAX 785 achieved the same quality results in less than 2 minutes. On a parallel machine, substantial further speedup will be possible.

In the new algorithm we no longer have the model order problem discussed in [2]. When a planar fit is applied to a step edge the estimates obtained from the two slope windows are both approximately zero.

5. Discussion

This short note has described a preliminary implementation of our algorithm, restricted to piecewise planar models. The extension of the technique to higher order polynomial fits as well as to other computer vision problems is a main focus of our current research.

We do not know the new estimator's breakdown point. The local variance based selection process uses less than half the data in the window to estimate the model parameters. We conjecture that the estimator can be modified to recover the model representing the *relative* and not the absolute majority of the data in the window.

It is important to note that the estimated model does not necessarily yield the minimum value of the median of squared residuals. Our goal in designing the algorithm was to develop a high breakdown point robust method for computer vision applications, not to preserve a particular optimality criterion.

References

- [1] M.A. Fischler and R.C. Bolles: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM* **24**, 381-395, 1981.
- [2] P. Meer, D. Mintz and A. Rosenfeld: Least median of squares based robust analysis of image structure. In these Proceedings.
- [3] P. Meer, D. Mintz, D.Y. Kim and A. Rosenfeld: Robust regression in computer vision: Least median of squares estimators. Revised manuscript

resubmitted to *International Journal of Computer Vision*, June 1990.

- [4] P. Meer and I. Weiss: Smoothed differentiation filters for images. In *Proceedings of the 10th International Conference on Pattern Recognition: Image, Speech and Signal Processing*. Atlantic City, NJ, June 1990, 474-479.
- [5] D.C. Montgomery and E.A. Peck: *Introduction to Linear Regression Analysis*. John Wiley & Sons, New York, 1982.

TABLE 1

Comparison Between Degree-0 LMedS
and the Computational Module
Used by the New Estimator

Noise	Estimator	Estimate	
		Mode	Spread
$\sigma = 0$	Degree-0 LMedS New Module	178	0
		178	0
$\sigma = 10$	Degree-0 LMedS New Module	176.5	1.5
		177.4	1.1
$\sigma = 25$	Degree-0 LMedS New Module	159.5	7.5
		176.4	3.2

Shape and Motion without Depth

Carlo Tomasi and Takeo Kanade

School of Computer Science

Carnegie Mellon University

Pittsburgh, Pa 15213

e-mail: ct@cs.cmu.edu, tk@cs.cmu.edu

Abstract

Inferring the depth and shape of remote objects and the complete camera motion from a sequence of images is possible in principle, but is an ill-conditioned problem when the objects are distant with respect to their size. We show how to overcome this problem by inferring shape and rotation without computing depth as an intermediate step.

On a single epipolar plane, an image sequence can be represented by the $F \times P$ matrix of the image coordinates of P points tracked through F frames. We show that under orthographic projection this matrix is of rank 3.

Using this observation, we develop an algorithm to recover shape and camera rotation, based on singular value decomposition. The algorithm gives accurate results, and does not introduce smoothing in either shape or motion.

1 Introduction

In principle, the shape of an object can be computed from a sequence of images by first estimating camera motion and depth, and then inferring shape from the depth values.

In practice, however, when objects are distant from the camera, relative to their size, this computation is ill-conditioned. First, the translation component along the optical axis is difficult to determine, because the image changes that it produces are small. Second, shape values are very sensitive to noise if they are computed as the small differences between large depth values.

These difficulties can be circumvented by inferring shape directly from variations in the relative position of image features, without computing depth as an intermediate step.

In this paper, we show that shape and camera rotation can be inferred precisely from many features and frames, without assuming any model for the motion, and reduce the computation to decomposing a matrix of image measurements.

The resulting algorithm, tested in simple situations, gives remarkably precise motion and shape estimates, without introducing smoothing effects into the result.

For simplicity, we will limit our consideration to one epipolar plane at a time, and assume that motion occurs in that plane. In other words, our images are single scanlines.

Our theory is based on the observation that the incidence relations among projection rays can be expressed as the degeneracy of a matrix that gathers all the image measurements. To our knowledge, this observation has not previously appeared in the literature.

Since we use many, closely spaced frames, the results are insensitive to noise, and the correspondence problem is simplified. Previous multi-frame approaches usually assume a motion model to combine estimates of the camera position over many frames. Typically, this model is some form of motion smoothness. In our method, on the other hand, we assume only the invariant of shape constancy over time.

As an illustration of our theory, we used our algorithm to recover the shape of a one-dollar silver coin (about 4 cm in diameter) placed at 3.5 meters from a real moving camera with a long lens. The total rotation of the camera was 30 degrees around the coin (and in the midplane of the coin). The error in the computed angle of camera rotation was always less than a tenth of one degree, and usually substantially smaller. The error in the shape of the coin was always less than 1.5 percent of its diameter, and typically considerably smaller. The small errors due to the effect of perspective are also analyzed.

In the following, we introduce our scenario, summarize the results, and sketch the relations of our work with previous literature on the subject. Section 2 introduces the degeneracy principle mentioned above. Section 3 shows how to use it to decompose the measurement matrix into shape and camera rotation. The experimental results in Section 4 show the ability of the algorithm to deal with jerky rotations without smoothing its output. The conclusion (Section 5) compares direct shape algorithms with algorithms that base the computation of shape on that of depth, and shows the former ones to be superior for remote scenes.

The Scenario

The world is still, and the camera moves in a plane, where it can freely rotate and/or translate. P feature points, far away from the camera, are visible in a given scanline, parallel to the plane of motion. Since the frames are taken frequently, it is easy to track the features from frame to frame. As the camera moves, it is panned so as to keep the features in the field of view.

After F frames, an $F \times P$ matrix U of image measurements is available. This matrix is the input to the algorithm.

This scenario approximates what happens with a camera on an airplane, with suitable control mechanisms to align the camera scanlines with the direction of flight, and to keep the same object within the field of view. Because objects

are distant from the camera, we can assume orthographic projection.

The Results

This paper first shows that if the measurements are noise-free, the image coordinate matrix U is highly degenerate: its rank is 3. As a result, U can be decomposed into the product of two smaller matrices: an $F \times 3$ matrix that encodes the F camera positions, and a $P \times 3$ matrix that encodes the positions of the P world points.

When noise corrupts the measurements, the rank of U can be defined in an approximate sense, and is still 3.

The noisy matrix U is factored by Singular Value Decomposition [Golub and Reinsch, 1971], which is known to be efficient and numerically well behaved. If more points and frames are used than prescribed by equation-counting arguments (which require a minimum of three points and three frames), the effects of noise can be reduced.

The resulting shape and motion algorithm is simple and efficient, and has been implemented and tested on objects as distant as one hundred times their size (see Section 4). The rotation errors are always smaller than one tenth of a degree. The relative precision in the computed shape is of the order of the *relative depth range*, defined as the ratio between the size of the object and its distance from the camera.

The good performance of our algorithm derives from the fact that shape is obtained directly, without using depth as an intermediate result. In traditional approaches, depth is first computed by triangulation. For remote objects, the quality of depth estimates by triangulation is very sensitive to noise, and degrades as the relative range decreases. Consequently, the shape estimates degrade even faster, since the computation of shape from depth is itself ill-conditioned.

In our approach, instead, no triangulation is done. Depth becomes irrelevant, and the results are highly accurate.

Relations with Previous Work

Our goal is to compute world point coordinates, relative to each other, and camera motion from multiple image frames.

Our algorithm does what photogrammetrists for more than thirty years have done by hand and with two frames at a time [Thompson, 1959]. Ullman proposed an automated solution to this problem eleven years ago [Ullman, 1979], and called it *structure-from-motion*. He also considered only two frames at once, and as few points as theoretically possible.

Most of the initial efforts in this area have been devoted to finding closed-form solutions with a minimal or nearly-minimal number of points and/or frames (see, for instance, [Longuet-Higgins, 1981]).

In general, structure-from-motion is hard to solve. The major difficulty is the inherent sensitivity of shape and motion to noise in the image, especially when objects are distant. If depth is explicitly represented as an intermediate stage in the computation, performance degrades with reductions in the relative depth range. For instance, the algorithm presented in [Tsai and Huang, 1984] works very well for close objects (which is the intended goal of that algorithm), but the performance is likely to degrade when objects become more remote, and the relative depth range becomes smaller.

The remedy is to by-pass the computation of depth, as we do in this paper, to remove the main cause of ill-conditioning.

Even with a well-conditioned algorithm, however, noise degrades performance. Few points and/or few frames give bad results, regardless of how good the math is. Our algorithm allows using many frames and many points, thus exploiting redundancy to counteract noise. If frames are closely spaced, the correspondence problem is also made easier to solve.

Many, tightly spaced frames have been used in [Bolles *et al.*, 1987] and [Matthies *et al.*, 1989], but only for the inference of depth when the motion of the camera is known. Determining shape and motion simultaneously, on the other hand, has been often suspected of being practically infeasible.

In [Spetsakis and Aloimonos, 1989], an interesting algorithm is presented for the case of unknown motion, using several frames and points and a perspective projection model. In spirit, our approach is akin to theirs: the projection lines of the same world point are a bundle (or pencil) of lines, and the resulting incidence relations between them allow casting the computation of shape and motion as a minimization problem. When applied to remote objects, however, their solution suffers from the same ill-conditioning problem discussed above, since depth is explicitly represented in their model.

2 The Decomposition Principle

This section introduces the fundamental principle on which our shape-and-motion algorithm is based: the $F \times P$ matrix of the image coordinates of P points tracked through F frames is highly rank-deficient.

As we stated in the introduction, we consider only one scanline per frame, and assume that the camera moves in a plane parallel to the scanline. In this plane, we define an arbitrary orthogonal system of coordinates (X, Z) .

The images are orthographic projections of P points, tracked through F frames. The registered measurements u_{fp} can then be collected in an $F \times P$ matrix

$$U = \begin{bmatrix} u_{11} & \cdots & u_{1P} \\ \vdots & & \vdots \\ u_{F1} & \cdots & u_{FP} \end{bmatrix}$$

From figure 1 we see that the projection u_{fp} of point p onto frame f is given by the equation

$$u_{fp} = c_f X_p + s_f Z_p + t_f, \quad (1)$$

where c_f and s_f are the cosine and sine of the angle α_f that frame f forms with the X axis. The scalar t_f is the projection onto the f -th image of the vector that joins the world origin with the origin of the f -th frame.

We can now collect all of the $F \times P$ equations (1) in matrix form:

$$U = MS \quad (2)$$

where

$$M = \begin{bmatrix} c_1 & s_1 & t_1 \\ \vdots & & \vdots \\ c_F & s_F & t_F \end{bmatrix} \quad (3)$$

is the motion matrix, and

$$S = \begin{bmatrix} X_1 & \cdots & X_P \\ Z_1 & \cdots & Z_P \\ 1 & \cdots & 1 \end{bmatrix} \quad (4)$$

is the shape matrix.

Since M is $F \times 3$ and S is $3 \times P$, we have just proven the following fact.

The Rank Principle

*Without noise, the rank of the measurement matrix U is at most three.*¹

Appendix A discusses the degenerate cases in which the rank of U is even smaller than three. These degeneracies correspond to all-aligned points or to special types of motion. They can always be detected, and treated as special cases. Consequently, we can simplify our treatment and assume that the rank principle is satisfied in a strong sense: the rank of U is exactly three.

Intuitively, the rank principle expresses the simple fact that the $F \times P$ image measurements are redundant. Indeed, they could all be described more concisely by giving F frame angles and P points, if only these were known.

Geometrically, the rank principle expresses an incidence property. In fact, if we replace X_p and Z_p in the projection equation (1) by the generic coordinates X and Z , we obtain the equation of the projection line of point p onto frame f :

$$u_{fp} = c_f X + s_f Z + t_f.$$

Equation (1) and, equivalently, the rank principle, say that there is a point that belongs to these lines for all values of f . In other words, the projection lines of a given point form a pencil.

In the next section, we show how to use the rank principle to determine the motion and shape matrices M and S .

3 The Algorithm

When noise corrupts the images, the measurement matrix U will not be exactly of rank 3. However, the rank principle can be extended to the case of noisy measurements in a well-defined manner. Subsection 3.1 introduces this extension, using the concept of Singular Value Decomposition (SVD) [Golub and Reinsch, 1971] to introduce the notion of approximate rank.

However, although the rank principle is the key to our algorithm, it is not the whole story. In Subsection 3.2, we show that, based on the rank principle, the matrices M and S are determined only up to an arbitrary affine warping of the plane. Therefore, in Subsection 3.2 we also point out the additional constraints needed to complete the solution.

Subsection 3.3 outlines the complete shape-and-motion algorithm.

3.1 Approximate Rank

Assuming² that $F \geq P$, the matrix U can be decomposed [Golub and Reinsch, 1971] into an $F \times P$ matrix L , a diagonal $P \times P$ matrix Σ , and a $P \times P$ matrix R ,

$$U = L \Sigma R^T, \quad (5)$$

such that

$$L^T L = R^T R = R R^T = I \\ \sigma_1 \geq \dots \geq \sigma_P.$$

Here, I is the $P \times P$ identity matrix, and the *singular values* $\sigma_1, \dots, \sigma_P$ are the diagonal entries of Σ . This is called the *Singular Value Decomposition* (SVD) of the matrix U .

We can now restate our key point.

The Rank Principle for Noisy Measurements

The first three singular values of the noisy measurement matrix U are much greater than the others:

$$\sigma_1, \sigma_2, \sigma_3 \gg \sigma_4, \dots, \sigma_P. \quad (6)$$

It can be shown [Forsythe *et al.*, 1977] that the rank-3 matrix U^* that is closest to U in the L_2 -norm sense can be obtained by setting to zero all the singular values after the third in the decomposition:

$$U^* = L^* \Sigma^* R^*, \quad (7)$$

where L^* collects the first three columns of L , Σ^* is the first third-order principal minor of Σ , and R^* gathers the first three rows of R .

3.2 The Metric Constraints

Golub and Reinsch [Golub and Reinsch, 1971] give an efficient and well-behaved algorithm to compute the singular value decomposition of a matrix. We use that algorithm to obtain a decomposition of the measurement matrix U .

The singular value decomposition of a matrix is unique because the left and right factors L and R are required to be orthonormal. However, this does not mean that there is only one way to decompose the measurement matrix U into M and S . Since the rank principle expresses an incidence relation, it only determines the two matrices M and S up to an affine transformation of the plane. In fact, if A is any invertible 3×3 matrix, the matrices MA and $A^{-1}S$ are also a valid decomposition of U , since

$$(MA)(A^{-1}S) = M(AA^{-1})S = MS = U.$$

Therefore, if we want to find M and S from the measurement matrix U , we need additional constraints. We approach the problem by first decomposing U into two matrices \hat{M} and \hat{S} of the appropriate sizes via the SVD algorithm. Based on equation (7), we can define, for instance, the two matrices

$$\hat{M} = L^*(\Sigma^*)^{1/2} \\ \hat{S} = (\Sigma^*)^{1/2}R^*. \quad (8)$$

Then, we can complete the solution by finding the matrix A that transforms \hat{M} and \hat{S} into the actual motion and shape matrices M and S :

$$M = \hat{M}A^{-1} \\ S = A\hat{S}.$$

The matrix A can be found by looking at the structure of the motion and shape matrices. The first and second column of M gather cosines and sines of the frame angles (see equation (3)), and must therefore be normalized. Furthermore, the third row of S contains all ones (equation (4)). These are *metric* constraints, as opposed to the incidence constraints expressed by the rank principle.

¹In [Tomasi and Kanade, 1990], all image coordinates were measured with respect to those of a reference feature. In that case, t_f was always zero, so the rank of the measurement matrix was two.

²This assumption is not crucial. If $F < P$, everything can be repeated for the transpose of U .

Formally, let us partition A and A^{-1} into rows and columns, respectively:

$$\begin{aligned} A &= \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \\ A^{-1} &= \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} B^T & b_3 \end{bmatrix}, \end{aligned} \quad (9)$$

where B^T gathers the first two columns b_1 and b_2 of A^{-1} . Then, the metric constraints above can be written as follows:

$$\begin{aligned} \hat{m}_f^T B^T B \hat{m}_f &= 1 \\ a_3 \hat{s}_p &= 1, \end{aligned} \quad (10)$$

where \hat{m}_f and \hat{s}_p are the f -th row and p -th column of \hat{M} and \hat{S} , respectively. These two equations say that the points \hat{m}_f are on a cylinder in a three-dimensional space, and that the points \hat{s}_p are on a plane in a three-dimensional space. The two equations are not independent, since a_3 and B^T are submatrices of A and A^{-1} , respectively. If we write out the product of A and A^{-1} as partitioned above, we see that the coupling can be expressed by the following equation:

$$a_3 B^T = \begin{bmatrix} 0 & 0 \end{bmatrix}. \quad (11)$$

Enforcing the pair of equations (10) leads to an overconstrained problem, and we can find the cylinder and the plane by data fitting.

In doing this, we encounter two difficulties. First, fitting a cylinder is a non-linear problem. Second, the two fitting problems are coupled through equation (11).

However, a well-behaved algorithm for our problem can be found by first determining a good approximation to the solution, and then refining the latter with a numerical function-minimization routine. This two-stage solution of the metric equations has proven to be accurate and robust in our experiments and simulations.

3.3 Outline of the Algorithm

The incidence and metric constraints expressed by the rank principle and by the cylinder and plane equations (10) are all we need for our algorithm. In conclusion, given an image measurement matrix U , the algorithm for computing the motion matrix M and the shape matrix S defined in equations (3) and (4) can be summarized as follows.

1. Compute the singular value decomposition of U :

$$U = L \Sigma R.$$

2. Define the initial decomposition of U into two matrices as follows:

$$\begin{aligned} \hat{M} &= L^* (\Sigma^*)^{1/2} \\ \hat{S} &= (\Sigma^*)^{1/2} R^*, \end{aligned}$$

where L^* collects the first three columns of L , Σ^* is the first third-order principal minor of Σ , and R^* gathers the first three rows of R .

3. Simultaneously fit a cylinder to the rows of \hat{M} and a plane to the columns of \hat{S} by minimizing the error criterion

$$e(a_3, B) = \sum_{f=1}^F (\hat{m}_f^T B^T B \hat{m}_f - 1)^2 + \sum_{p=1}^P (a_3 \hat{s}_p - 1)^2$$

subject to the constraint

$$a_3 B^T = \begin{bmatrix} 0 & 0 \end{bmatrix}.$$

4. Complete the matrix A and its inverse from their submatrices a_3 and B by solving the system

$$A A^{-1} = I.$$

5. Compute the motion matrix M and the shape matrix S as

$$\begin{aligned} M &= \hat{M} A^{-1} \\ S &= A \hat{S}. \end{aligned}$$

The details of the fitting algorithm in step 3 and of the matrix completion of step 4 are described in appendices B and C, respectively.

4 An Experiment

We implemented the algorithm described in the previous section, and applied it on several image sequences.

The experiment described in this section illustrates the rank principle, demonstrates the good quality of the results, and quantifies the influence of perspective effects on the accuracy of the motion estimates.

The key parameter for the evaluation of performance is the relative depth range, which we defined as the ratio of the object size along the optical axis and the distance between camera and object. In a nutshell, the conclusion drawn from our experiments is that the relative errors in the computed shape are of the same order as the relative depth range. Consequently, modeling inaccuracies that are small with respect to the latter can be ignored.

We put a one-dollar coin (about 4 cm in diameter) approximately 3.5 meters away from a Sony CCD camera with a 300 mm Tokina lens. Thus, the relative depth range was $4/350 \approx 0.011$. Figure 2 shows the setup.

The camera was moved in the plane of the coin, so that only the edge of the coin was visible in every frame. The motion was roughly circular around a point in the vicinity of the coin. Only the rotation component was controlled with an accurate positioning mechanism, so that precise ground truth was available for performance evaluation. Translation was such as to keep the coin in the field of view, but was otherwise uncontrolled.

The edge of the coin was approximately aligned with the image scanlines, thus yielding easy-to-track image features (the thin vertical notches on the coin's edge). The first 101 frames were taken in steps of 0.1 degrees between consecutive frames. After that, the velocity was doubled to 0.2 degrees per frame, and 100 more frames were taken. Thus, the overall rotation was 30 degrees. The resulting 201 scanlines are stacked together in figure 3, top to bottom. This figure is what is called an epipolar plane in [Bolles *et al.*, 1987].

The image was filtered with a thirteen-tap finite-impulse-response approximation to the Laplacian of a Gaussian, and the 104 zero crossings of the result, shown in figure 4, were used as features in the experiment.

The measurement matrix was thus 201×104 in size. All of the processing, including feature extraction and linking,

matrix decomposition, and motion and shape computation, took about three minutes on a VAX 8800.

The rank principle is illustrated graphically by the similarity of figures 4 and 5. To obtain figure 5, we decomposed the matrix U representing the crossings, set to zero all the singular values except the first three, and reconstructed the measurement matrix. Thus, figure 5 represents the rank-3 matrix U^* of equation (7). The rank principle says that the only differences between figure 4 and figure 5, under orthography, are due to noise.

The singular values are plotted in figure 6; without noise, and if the projection were exactly orthographic, only the first three values would be different from zero.

Figure 7 shows the computed and the true rotation. The error is always smaller than one tenth of one degree, and almost everywhere substantially smaller than that. The algorithm assumes no motion models, and does no smoothing. As a result, the sharp change in rotational velocity after frame 100 is faithfully preserved in the motion output.

Figure 8 shows the shape results, and the best circular fit to them. The accuracy of shape is of the order of the relative depth range (1 percent), even if variations in depth during the motion of the camera were of the order of the coin size.

In spite of image noise, perspective effects and unmodeled small variations in depth, the quality of both shape and motion results is remarkably good.

To get an idea of how perspective effects influence the accuracy of the results, we tested our algorithm on a sequence of simulated, noise-free images similar to those of our coin experiment. A circular object with 10 features is placed at various depths from the camera. For each depth, a pinhole camera moves and rotates by 30 degrees in 30 steps. Figure 9 plots the relative error in the total computed rotation as a function of the relative depth range. While algorithms based on depth give worse motion estimates as objects are moved farther away, our algorithm improves (for a constant total rotation angle), because it approximates orthography better and better.

5 Conclusion: Depth versus Shape Algorithms

The algorithm presented in this paper infers the shape of remote objects and the rotation of the camera. It is a *shape* algorithm. It does not compute the depth of the scene.

Algorithms such as the ones described in [Tsai and Huang, 1984], [Heel, 1989], [Spetsakis and Aloimonos, 1989], on the other hand, represent depth explicitly, and compute it from the image sequence. They are *depth* algorithms.

Depth algorithms give a more complete answer. They compute all components of motion, up to a scale factor, and the depth information they supply allows, in principle, computing shape as well.

However, depth algorithms do not work if objects are very distant from the camera with respect to their size. When the relative depth range is very small, as for instance in aerial cartography and reconnaissance, the completeness of depth algorithms is not only useless, but harmful. A shape algorithm gives a more stable and accurate answer, because it computes shape and camera rotation directly from image deformations, without using depth as an intermediate step.

The results of this paper can be extended in many ways: accuracy, three-dimensionality, completeness, efficiency.

Accuracy can be increased even further by correcting for perspective effects. Once a good shape estimate has been computed, the solution can be perturbed with a steepest descent search to account for the slight divergence of projection rays in each frame. Furthermore, if depth varies dramatically, looming effects must be accounted for.

The algorithm can be extended to three dimensions. For obvious reasons of applicability, this is the direction we have chosen to pursue next in our research.

Completeness: if a motion model is available, depth and camera translation can be estimated independently. Shape and rotation, computed by our algorithm, would be *inputs* to a separate depth and translation algorithm, possibly together with external motion information. Shape and depth are often several orders of magnitude apart. We have shown that they should be computed separately, not that depth cannot be estimated.

Our implementation of the algorithm uses an efficient singular-value-decomposition routine. However, it treats a whole batch of frames at once. An incremental implementation would be more desirable. The feasibility of this is being investigated.

References

- [Bolles *et al.*, 1987] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1(1):7-55, 1987.
- [Forsythe *et al.*, 1977] G. E. Forsythe, M. Malcolm, and C. B. Moler. *Computer Methods for Mathematical Computations*. Prentice-Hall, Englewood Cliffs, NJ, 1977.
- [Golub and Reinsch, 1971] G. H. Golub and C. Reinsch. *Singular Value Decomposition and Least Squares Solutions*, volume 2, chapter I/10, pages 134-151. Springer Verlag, New York, NY, 1971.
- [Heel, 1989] J. Heel. Dynamic motion vision. In *Proceedings of the DARPA Image Understanding Workshop*, pages 702-713, Palo Alto, CA, May 23-26 1989.
- [Longuet-Higgins, 1981] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133-135, September 1981.
- [Matthies *et al.*, 1989] L. Matthies, T. Kanade, and R. Szeliski. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3(3):209-236, September 1989.
- [Spetsakis and Aloimonos, 1989] M. E. Spetsakis and J. Y. Aloimonos. Optimal motion estimation. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 229-237, Irvine, California, March 1989.
- [Thompson, 1959] E. H. Thompson. A rational algebraic formulation of the problem of relative orientation. *Photogrammetric Record*, 3(14):152-159, 1959.
- [Tomasi and Kanade, 1990] C. Tomasi and T. Kanade. Shape and motion without depth. Technical report CMU-CS-90-128, Carnegie Mellon University, Pittsburgh, PA, May 1990.

[Tsai and Huang, 1984] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1):13-27, January 1984.

[Ullman, 1979] S. Ullman. *The Interpretation of Visual Motion*. The MIT Press, Cambridge, MA, 1979.

A Degeneracies of the Measurement Matrix

If both the scene and the camera motion are sufficiently complex, the measurement matrix U is exactly of rank 3. On the other hand, special object shapes and/or particular types of camera motion can further reduce the rank of U .

In this section we show that the object shape is degenerate if and only if all feature points are aligned, and that camera motion is degenerate if and only if it is such that all optical axes pass through the same point.

Shape degeneracies

We now interpret the determinants

$$\Delta_{fg}^{(pq)} = \det \begin{bmatrix} u_{fp} & u_{fq} \\ u_{gp} & u_{gq} \end{bmatrix}$$

in terms of intrinsic geometric parameters which describe the relative position of the three world points, and of the angles between frames.

It follows immediately from this interpretation that a necessary and sufficient condition for the existence of at least one non-zero determinant of the type above is that there be at least three non-aligned points, and at least two distinct frames.

Since we consider only shape degeneracies, we can set $t_f = 0$ for all f . This is equivalent to saying that the camera moves by pure rotation, as shown in figure 10.

Let d_p and γ_p be the magnitude and phase of the vector which joins the center of rotation, chosen as the world origin, with object point number p :

$$\begin{aligned} d_p &= \sqrt{X_p^2 + Z_p^2} \\ \gamma_p &= \arctan_2(Z_p, X_p) \end{aligned}$$

(see figure 10).

Here \arctan_2 is the two-argument inverse tangent function, which differs from the one-argument function in that it returns the angle in the appropriate quadrant, and has no singularities:

$$\arctan_2(y, x) = \begin{cases} \arctan(y/x) & \text{if } x > 0 \\ \text{sign}(y)(\pi - \arctan(|y/x|)) & \text{if } x < 0 \\ 0 & \text{if } x = y = 0 \\ \text{sign}(y)\pi/2 & \text{if } x = 0, y \neq 0 \end{cases}$$

Furthermore, let ψ_{fg} be the angle between frame f and frame g , measured counterclockwise from f to g (figure 10), and let $\gamma_{pq} = \gamma_p - \gamma_q$.

Then, if u_{fp} is the projection of point p onto frame f , we have

$$\Delta_{fg}^{(pq)} = \det \begin{bmatrix} u_{fp} & u_{fq} \\ u_{gp} & u_{gq} \end{bmatrix} = d_p d_q \sin \gamma_{pq} \sin \psi_{fg}.$$

Proof

We introduce the angles ω_{fp} between frame f and the line from the origin to point p ; the determinant $\Delta_{fg}^{(pq)}$ is easily expressed in terms of these angles:

$$\begin{aligned} \Delta_{fg}^{(pq)} &= \det \begin{bmatrix} d_p \cos \omega_{fp} & d_q \cos \omega_{fq} \\ d_p \cos \omega_{gp} & d_q \cos \omega_{gq} \end{bmatrix} \\ &= d_p d_q (\cos \omega_{fp} \cos \omega_{gq} - \cos \omega_{fq} \cos \omega_{gp}) \\ &= \frac{d_p d_q}{2} [\cos(\omega_{fp} + \omega_{gq}) + \cos(\omega_{fp} - \omega_{gq}) \\ &\quad - \cos(\omega_{fq} + \omega_{gp}) - \cos(\omega_{fq} - \omega_{gp})]. \end{aligned}$$

If we now observe that

$$\begin{aligned} \omega_{fq} &= \omega_{fp} + \gamma_{pq} \\ \omega_{fp} &= \psi_{fg} + \omega_{gp} = \psi_{fg} + \omega_{gq} - \gamma_{pq} \end{aligned}$$

we can write

$$\begin{aligned} \omega_{fp} + \omega_{gq} &= \omega_{fq} + \omega_{gp} = 2\omega_{fp} - \psi_{fg} + \gamma_{pq} \\ \omega_{fp} - \omega_{gq} &= \psi_{fg} - \gamma_{pq} \\ \omega_{fq} - \omega_{gp} &= \psi_{fg} + \gamma_{pq}, \end{aligned}$$

so that

$$\begin{aligned} \Delta_{fg}^{(pq)} &= \frac{d_p d_q}{2} [\cos(\psi_{fg} - \gamma_{pq}) - \cos(\psi_{fg} + \gamma_{pq})] \\ &= d_p d_q \sin \gamma_{pq} \sin \psi_{fg}, \end{aligned}$$

as promised.

Motion degeneracies

The motion matrix M defined in equation (3) is of rank smaller than three if and only if one column is a linear combination of the other two. We consider separately two cases, depending on whether the two vectors $\mathbf{c} = (c_1 \dots c_F)^T$ and $\mathbf{s} = (s_1 \dots s_F)^T$ are mutually dependent.

The vectors \mathbf{c} and \mathbf{s} are dependent only when all inter-frame rotations are integer multiples of $\pi/4$. The only interesting case of this type occurs when the camera moves by pure translation. In this case, all optical axes pass through the same point at infinity.

If, on the other hand, \mathbf{c} and \mathbf{s} are mutually independent, the motion matrix M (and therefore the measurement matrix U) is of rank two if and only if there are two numbers α and β such that

$$t_f = \alpha c_f + \beta s_f. \quad (12)$$

For a generic point (\hat{X}, \hat{Z}) , the projection equation (1) can be rewritten in the following form:

$$t_f = u_{fp} - \hat{X} c_f - \hat{Z} s_f.$$

By comparing this equation with equation (12), we see that for the latter to hold there must be a (possibly invisible) point with coordinates $\hat{X} = -\alpha$ and $\hat{Z} = -\beta$ that is always projected to the origin, that is, such that $u_{fp} = 0$.

Since the projection ray of a point that projects to the origin is the optical axis, this proves that motion is degenerate if and only if all optical axes pass through the same point.

B Simultaneous Cylinder and Plane Fitting

This Appendix elaborates on step 3 of our algorithm: the minimization of the error criterion

$$e(a_3, B) = e_m(B) + e_s(a_3)$$

where

$$e_m(B) = \sum_{f=1}^F (\hat{\mathbf{m}}_f^T B^T B \hat{\mathbf{m}}_f - 1)^2$$

and

$$e_s(a_3) = \sum_{p=1}^P (a_3 \hat{\mathbf{s}}_p - 1)^2$$

subject to the constraint

$$\mathbf{a}_3 B^T = \begin{bmatrix} 0 & 0 \end{bmatrix}. \quad (13)$$

We compute the solution in the following steps:

- find the cylinder $\hat{B}^T \hat{B}$ that minimizes $e_m(B)$
- find the plane $\hat{\mathbf{a}}_3$ that minimizes $e_s(a_3)$
- minimize $e(a_3, B)$ numerically, using $\hat{\mathbf{a}}_3$ and \hat{B} as a starting point, and enforcing the constraint of equation (13).

We now examine the first and the third step in some detail. The second step, fitting a plane, is trivial.

Fitting the Cylinder

Fitting a cylinder $\mathbf{m}^T B^T B \mathbf{m} = 1$ to a set of three-dimensional data $\hat{\mathbf{m}}_1, \dots, \hat{\mathbf{m}}_F$ is a non-linear problem in the entries of the matrix B .

Consequently, we use the same strategy as above: we first find a good approximation to the solution, and then we refine it numerically.

The approximation can be found by first fitting a quadratic form $\mathbf{m}^T Q \mathbf{m} = 1$ to the data. Thus, rather than finding a cylinder, we find an ellipsoid. This is a linear problem in the entries of the symmetric matrix Q , and can be solved easily. We then decompose the result, \hat{Q} , and set its smallest eigenvalue to zero. The decomposition yields a first approximation to \hat{B} . In this way, instead of finding the optimal cylinder, we obtain the cylinder that is closest to the optimal ellipsoid. From there, we can reach the optimal cylinder by numerical minimization of $e_m(B)$. Our experiments indicate that this last step is hardly necessary: the cylinder obtained by suppressing the smallest eigenvalue of \hat{Q} is almost the same as the optimal cylinder.

Refining the Minimum of $e(a_3, B)$

We now have a cylinder $\hat{B}^T \hat{B}$ and a plane $\hat{\mathbf{a}}_3$ which separately minimize the two error functions $e_m(B)$ and $e_s(a_3)$. However, \hat{B} and $\hat{\mathbf{a}}_3$ may not satisfy the constraint (13) exactly.

In order to enforce equation (13), and at the same time minimize the global error function $e(a_3, B)$, we use the constraint to write \mathbf{a}_3 as a function of B . Equation (13) says that \mathbf{a}_3 is orthogonal to both rows, \mathbf{b}_1 and \mathbf{b}_2 , of B , so we can write

$$\mathbf{a}_3 = a_3(\mathbf{b}_1 \times \mathbf{b}_2)$$

where \times denotes the cross product, and a_3 is a scalar.

As a result, we obtain a function $e'(a_3, B)$ of only seven variables, rather than nine. The minimization of e' is now unconstrained.

To complete the task we need the derivatives of the cylinder residue function $\gamma = \mathbf{m}^T B^T B \mathbf{m} - 1$ and the plane residue function $\pi = a_3(\mathbf{b}_1 \times \mathbf{b}_2) \cdot \mathbf{s} - 1$ with respect to the unknown parameters a_3 and B , for use in a standard minimization routine.

Simple algebraic manipulation shows the derivatives to be

$$\frac{\partial \gamma}{\partial B} = 2B\mathbf{m}\mathbf{m}^T$$

(a 2×3 matrix of derivatives), and

$$\begin{aligned} \frac{\partial \pi}{\partial \mathbf{b}_1} &= a_3(\mathbf{b}_2 \times \mathbf{s}) \\ \frac{\partial \pi}{\partial \mathbf{b}_2} &= -a_3(\mathbf{b}_1 \times \mathbf{s}) \\ \frac{\partial \pi}{\partial a_3} &= \det \begin{bmatrix} \mathbf{s} \\ B \end{bmatrix}. \end{aligned}$$

C Completion of the Matrix A and its Inverse

This Appendix shows how to complete the matrix

$$A = \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{bmatrix}$$

and its inverse

$$A^{-1} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 \end{bmatrix} = \begin{bmatrix} B^T & \mathbf{b}_3 \end{bmatrix}$$

given their submatrices \mathbf{a}_3 and B . This is step 4 of our algorithm.

The 3×3 matrix equation

$$AA^{-1} = I$$

can be expanded into nine scalar equations:

$$\begin{aligned} \mathbf{a}_1 \mathbf{b}_1 &= 1 & \mathbf{a}_1 \mathbf{b}_2 &= 0 & \mathbf{a}_1 \mathbf{b}_3 &= 0 \\ \mathbf{a}_2 \mathbf{b}_1 &= 0 & \mathbf{a}_2 \mathbf{b}_2 &= 1 & \mathbf{a}_2 \mathbf{b}_3 &= 0 \\ \mathbf{a}_3 \mathbf{b}_1 &= 0 & \mathbf{a}_3 \mathbf{b}_2 &= 0 & \mathbf{a}_3 \mathbf{b}_3 &= 1. \end{aligned}$$

The two equations

$$\mathbf{a}_3 \mathbf{b}_1 = 0 \quad \text{and} \quad \mathbf{a}_3 \mathbf{b}_2 = 0$$

contain only known quantities. They coincide with the constraint equation (13), and can be ignored here. Since the unknown scalars are still nine (the entries of \mathbf{a}_1 , \mathbf{a}_2 , and \mathbf{b}_3), we need two more equations.

These two degrees of freedom derive from the fact that the origin of the world coordinate system was left unspecified. Rather than constraining the origin to be at (0,0), we use these degrees of freedom to improve the noise performance of the shape result as follows.

The shape matrix is computed as $S = A\hat{S}$ in the last step of our algorithm. Of the three columns of \hat{S} , the third is the most sensitive to noise, because it corresponds to the smallest singular value of the decomposition (5). Consequently, it is advantageous to avoid using that column in the final result. This can be accomplished by requiring the third entries of \mathbf{a}_1 and \mathbf{a}_2 to be zero:

$$\begin{aligned} \mathbf{a}_1 \mathbf{v} &= 0 \\ \mathbf{a}_2 \mathbf{v} &= 0, \end{aligned}$$

where $v = (0, 0, 1)^T$.

We now have the nine equations we need. The six homogeneous equations express orthogonality, and we can use them to find the directions (unit vectors) w_1, w_2 , and w_3 of the unknown vectors a_1, a_2 , and b_3 . From $a_1 b_2 = 0$ and $a_1 v = 0$ we deduce that a_1 is orthogonal to both b_2 and v , so that its unit vector is

$$w_1 = \frac{b_2 \times v}{|b_2 \times v|}.$$

Similarly, for the unit vector of a_2 , the two equations $a_2 b_1 = 0$ and $a_2 v = 0$ yield

$$w_2 = \frac{b_1 \times v}{|b_1 \times v|}.$$

From these two results, and equations $a_1 b_3 = 0$ and $a_2 b_3 = 0$, we obtain the unit vector of b_3 :

$$w_3 = \frac{w_1 \times w_2}{|w_1 \times w_2|}.$$

The signed magnitudes α_1, α_2 , and β_3 of a_1, a_2 , and b_3 can now be found from the non-homogeneous equations

$$\begin{aligned} a_1 b_1 &= 1 \\ a_2 b_2 &= 1 \\ a_3 b_3 &= 1, \end{aligned}$$

which yield

$$\begin{aligned} \alpha_1 &= 1/(\beta_1 \cos \theta_1) \\ \alpha_2 &= 1/(\beta_2 \cos \theta_2) \\ \beta_3 &= 1/(\alpha_1 \cos \theta_3), \end{aligned}$$

where $\beta_1, \beta_2, \alpha_3$ are the magnitudes of the known vectors b_1, b_2, a_3 , and $\theta_1, \theta_2, \theta_3$ are the angles between a_1 and b_1, a_2 and b_2, a_3 and b_3 , that is,

$$\begin{aligned} \cos \theta_1 &= w_1 \frac{b_1}{|b_1|} \\ \cos \theta_2 &= w_2 \frac{b_2}{|b_2|} \\ \cos \theta_3 &= w_3 \frac{a_3}{|a_3|}. \end{aligned}$$

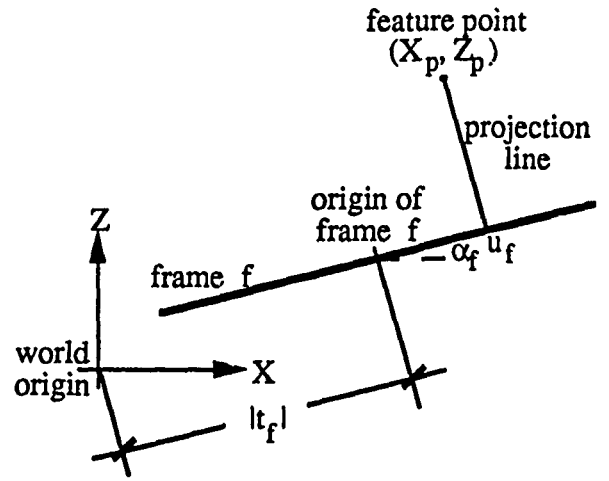


Figure 1: The basic geometry.

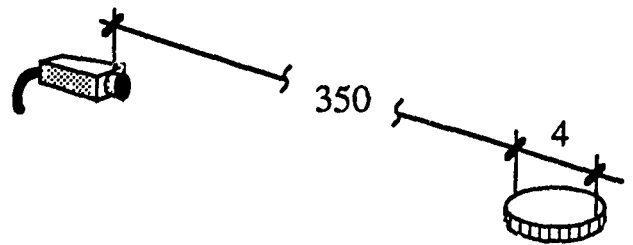


Figure 2: The setup in our experiment. Measures are in centimeters.

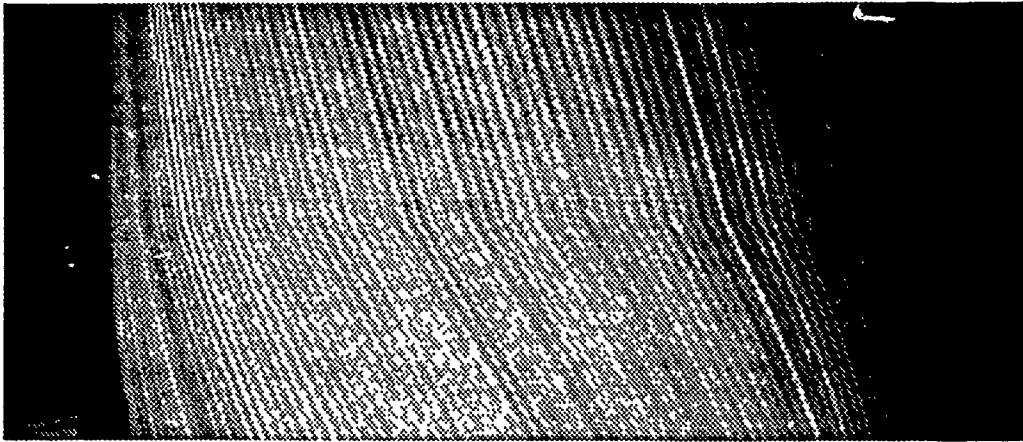


Figure 3: The input to the algorithm; each scanline is a new frame, and represents the edge of a one-dollar coin seen from a new angle. In [Bolles *et al.*, 1987], a figure like this is called an epipolar plane. We use it to recover shape and rotation, instead of depth given known motion.

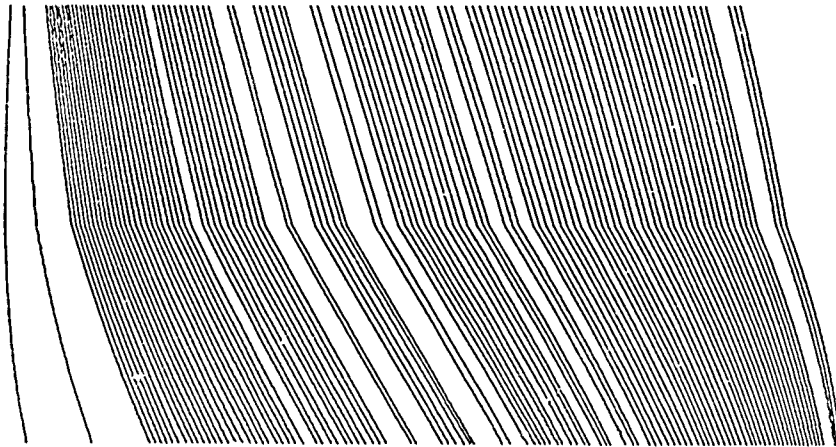


Figure 4: The zero crossings from figure 3.

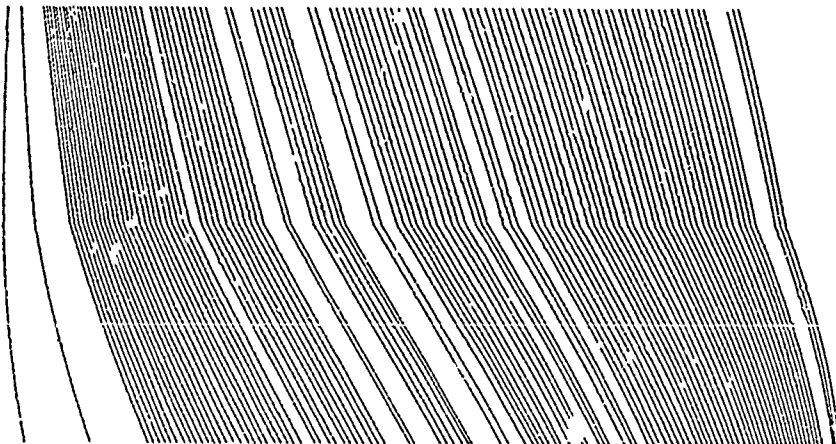


Figure 5: Zero crossings reconstructed after suppressing all but the first three singular values of the measurement matrix.

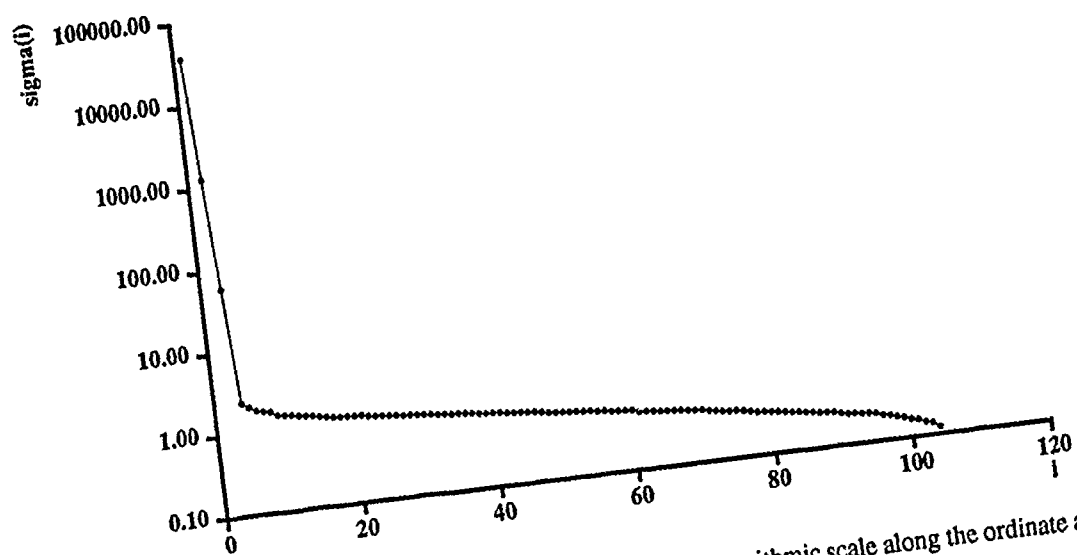


Figure 6: Singular values of the measurement matrix. Notice the logarithmic scale along the ordinate axis.

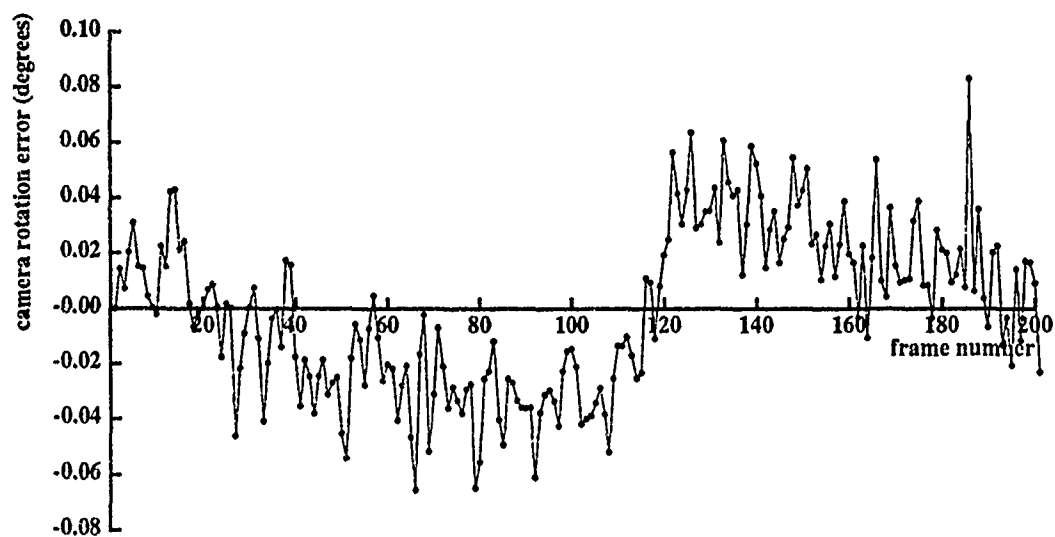
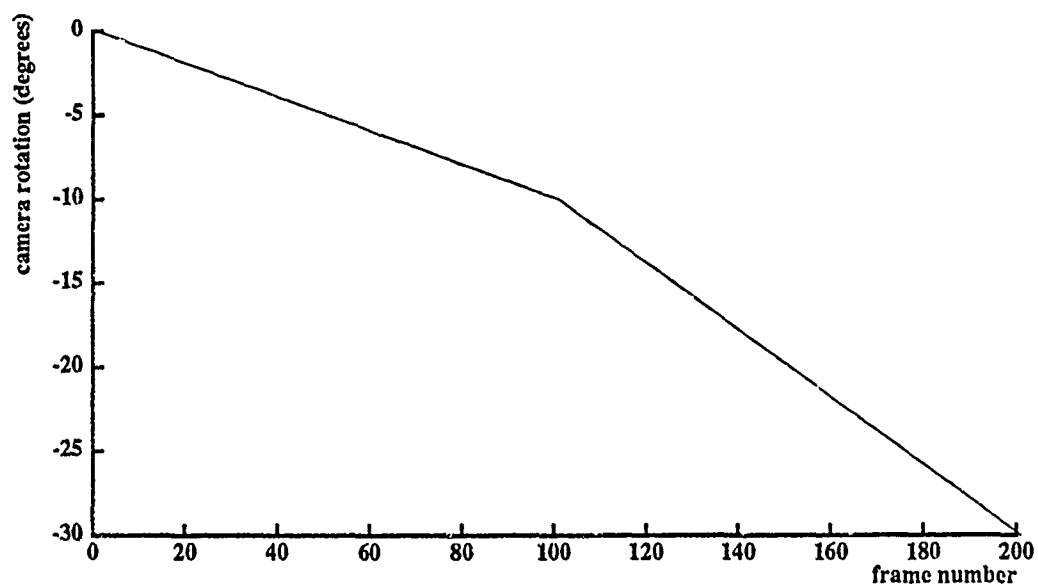


Figure 7: Camera rotation. In the top plot, both the computed (solid) and the true (dashed) rotation are plotted, but the difference is so small that they can hardly be distinguished. In the plot below, the difference between the two graphs is enlarged.

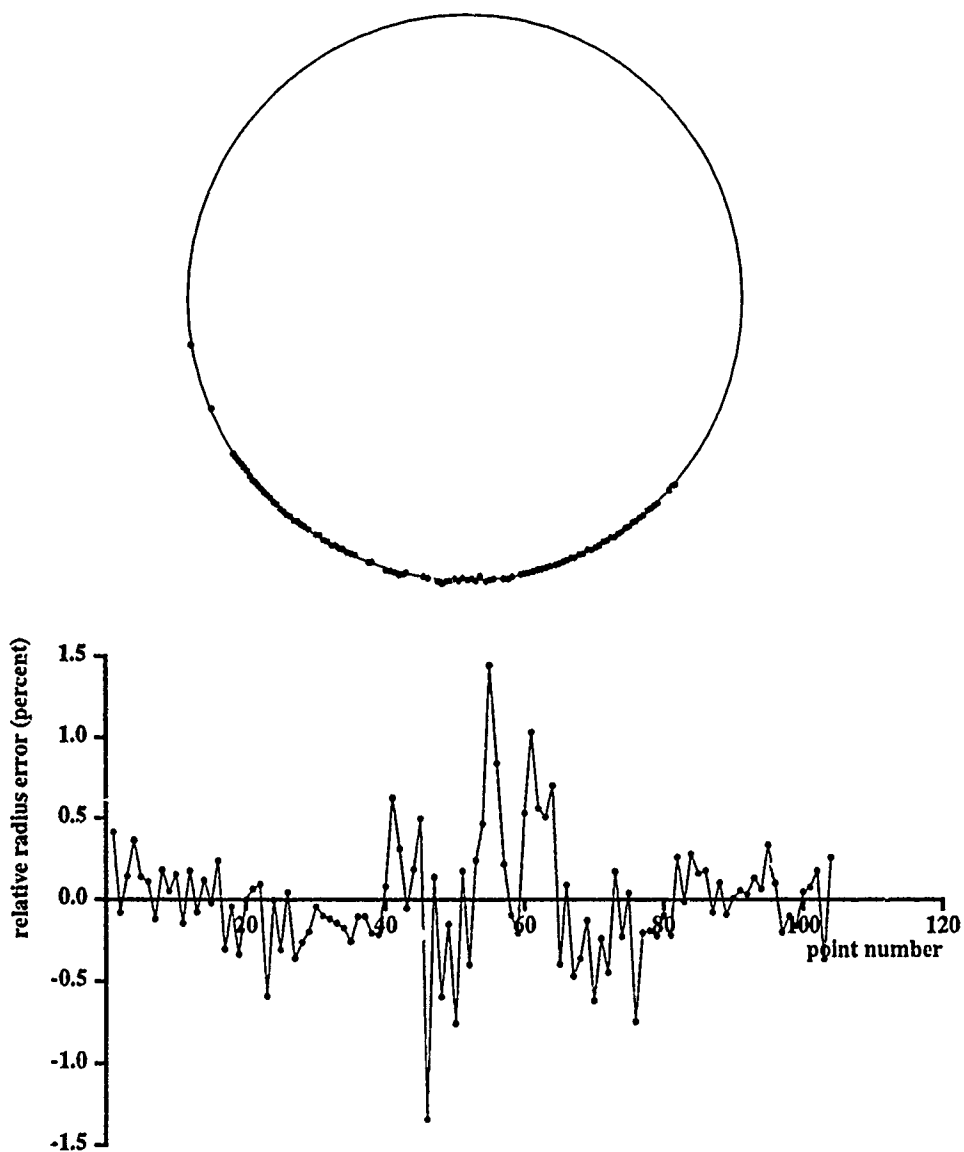


Figure 8: Shape. The top figure shows the computed shape (dots) of a one-dollar coin, with the best fit circle. The bottom figure magnifies the difference between true and computed shape values along the radius of the coin.

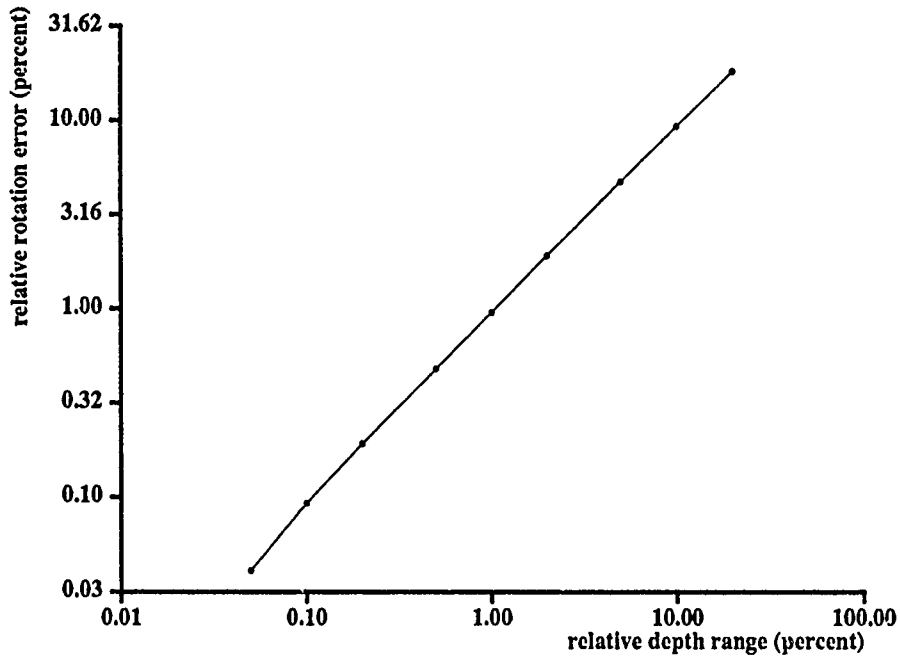


Figure 9: The motion error due to perspective distortion decreases when the relative depth range becomes smaller. These results were obtained by simulating noise-free images of a circular object with 10 features, and a pin-hole camera rotating by 30 degrees in 30 frames.

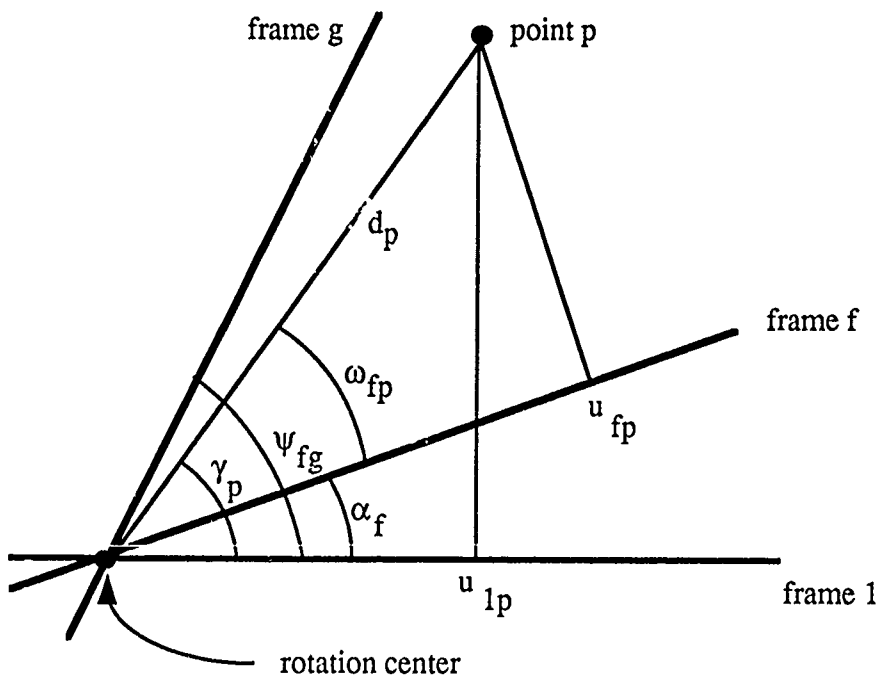


Figure 10: The angles defined in appendix A.

A Unified Theory of Structure from Motion

Minas E. Spetsakis* John (Yiannis) Aloimonos
Computer Vision Laboratory, Center for Automation Research,
University of Maryland, College Park, MD 20742-3411

ABSTRACT

The long sought linear algorithm for the point and line correspondence problem is presented in the first section of this paper. Next a new statistical definition of feature points is introduced, under which point features and line features are just the two extremes of a spectrum of possible features. Almost any pixel in the image can be classified and used as a feature point in this scheme. Based on this definition we have designed an optimal algorithm for the structure from motion problem that can utilize information from across the whole image. The input to the algorithm is the image displacement, and its uncertainty, at each pixel for a set of three frames. The only assumptions used are rigidity and Gaussian noise in the image displacements. The outputs are the parameters of the motion between the frames and the structure of the scene.

The theory behind our approach is simple and elegant, can be extended in several ways (e.g. to multiple frames), and is developed with noise stability in mind. However, more important is that the new statistical definition of the features relaxes the requirements for the image displacement computation. In fact, if the tangential component of a displacement cannot be computed then its uncertainty is set to infinity, and the algorithm can tolerate infinite uncertainty for all the tangential components. In this way the aperture problem is avoided.

1 Geometry and Statistics

When one looks at a real image one can see a number of identifiable features such as points and lines. It has been argued that these features are the only things needed for computing the motion and structure because they carry reliable information, there exist mathematical and computational tools to treat them, and extensive experience and literature from photogrammetry can be tapped. On the other hand the number of image pixels that are covered by these features is only a tiny fraction of the total number of

pixels in the image. If one uses only point and line features, the vast majority of the image remains unused. The feature points carry more information than the rest of the pixels, but the rest of the pixels are much more numerous. They should not be left unused. This underutilization is not the only problem; two additional, more solid problems arise. First, there is no consensus among researchers in computer vision on what a feature point or line is, either in a rigorous mathematical sense or even in an intuitive practical sense, judging from what different detectors detect as features. The main consequence of this is that there is no general algorithm to detect features and match them. Second, even if one can detect point and line features, there exists no algorithm that works with both of them at the same time and guarantees a unique solution, although there are algorithms for each one separately [Longuet-Higgins, 1981; Spetsakis and Aloimonos, 1987; Tsai and Huang, 1984]. A special case of the theory presented here is an algorithm that can treat both of them at the same time.

There is another approach to structure from motion that assumes continuous motion and grey level images that are differentiable in time and space. Despite its theoretical elegance, this approach is plagued by the aperture problem: the motion (optic flow) of a point on a moving curved line (isophote, zero crossing, etc.) cannot be recovered fully; we can recover only its projection on the normal to the line. Most algorithms based on this approach assume that derivatives up to second order of the (not fully known) image flow are given or can be computed, and that the flow is smooth. The resulting algorithms are local (and hence unstable) in nature [Longuet-Higgins and Prazdny, 1980; Waxman and Worn, 1987].

Obviously there is need to overcome these difficulties and combine the advantages of all the existing methods. This does not seem to be an easy task within the existing theories, since they all are rather incompatible, with different input requirements and conflicting assumptions, and they operate on different geometric entities. But are points, lines, curves or isophotes different entities or can they be defined in a uniform way? The truth is that they are different when compared as abstract geometric entities. In the context of visual motion a feature point is a small area of the image that (besides the statistics of its grey level that made the detector locate it) is moving with a motion whose uncertainty is more or less circularly symmetric and small (finite). In the same context, a line is an area whose motion has an uncertainty that is

*Current address: Department of Computer Science, York University, 4700 Keele Street, North York, Ontario, Canada M3J 1P3.

The support of the Defense Advanced Research Projects Agency (ARPA Order No. 6989) and the U.S. Army Engineer Topographic Laboratories under Contract DACA76-89-C-0019 is gratefully acknowledged.

finite in one direction (normal to the line) and infinite in the other direction (along the line). These definitions, having an obvious statistical flavor, seem natural and uniform. We can forget the separate definitions of points and lines and concentrate instead on the statistics of the disparity fields. The only geometric entity we need is the curve in its most general sense, whether it be a chain of edgels, a zero crossing, or an isophote (Figure 1). Thus any point in the image can be considered. If such a curve is moving then a point that was on it in the first frame will move with the curve but we do not know where on the length of the curve it will be in the next frame. We only know that it is on the curve but we have only a probability distribution for where it can be on the curve. This probability distribution accounts for the tangential component of the uncertainty of the motion of the point. There is also another component of uncertainty, the normal one, because due to the fuzziness of the curve we cannot assume that the motion of the point along the normal direction can be recovered exactly. Thus we have a probability distribution along both directions. So we do not assume that we shall be given the exact displacement vector of any point but only a probability distribution for it. The estimation of this distribution is not within the scope of this paper. As a working assumption we will assume that this distribution is Gaussian and its parameters are given. Gaussian is a very good choice because it makes sense intuitively and leads to very stable statistics, meaning that if this assumption holds only approximately then the consequences are not catastrophic and the degradation is graceful. Also, coupled with maximum likelihood it gives rise to least-squares estimators which have nice analytic expressions.

An $n \times 1$ vector p (either position or displacement vector) follows a Gaussian distribution with joint variance A and mean μ when its probability density function is

$$f(p) = \frac{1}{\sqrt{(2\pi)^n \cdot \det(A)}} e^{-(p-\mu)^T \cdot A^{-1} \cdot (p-\mu)}$$

A is an $n \times n$ square matrix and μ an $n \times 1$ vector. In Figures 1 and 2, in the areas where the probability density is significant we have drawn a cloud of dots. These clouds have ellipsoidal shapes the main axes of which are the eigenvectors of matrix A . The eigenvalues of A represent the variances along the corresponding main axes. In Figure 2 the meaning of points and lines is given in these terms.

One final word about this framework. Part of the difficulty in structure from motion is its absolute separation from the preceding stage of computing the displacement vector field (or flow field or correspondence; they all are of the same flavor). This unavoidably leads to the idea of trying to find the exact displacement field using restrictive assumptions such as smoothness [Horn and Schunck, 1981] and then, pretending that this is the correct field, find the structure and motion [Spetsakis and Aloimonos, 1988; Subbarao and Waxman, 1985; Tsai and Huang, 1984]. Here instead we require much less from the preceding step than a complete, accurate disparity field, thus eliminating the complete reliance on assumptions such as smoothness. Then, using the theory described below, the only assumptions of which are rigidity and Gaussian noise, we find the motion and structure (with some uncertainty depending on the data). If indeed the error in the data follows the Gaussian distribution, then the structure and motion we compute is the optimal one. Otherwise it is not, and what we have computed is a good approximation which is the collective result of information from a large area of the image. Once we have this collective result, which is a constraint on the flow (by backprojection), we can couple it with the original gray level based constraint and compute the displacement field again. It seems

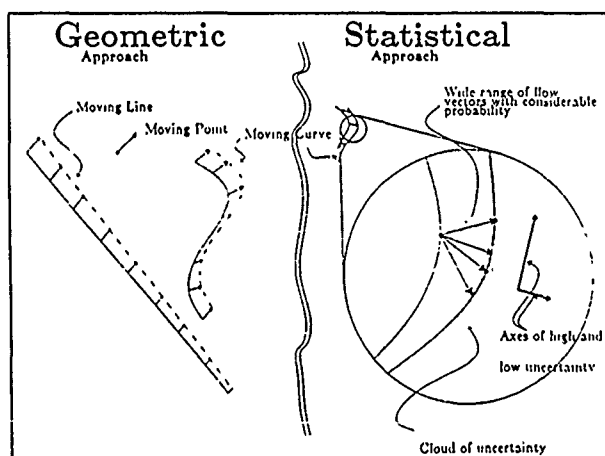


Figure 1. We need to redefine the meaning of features so that almost any pixel carrying some information can fit in.

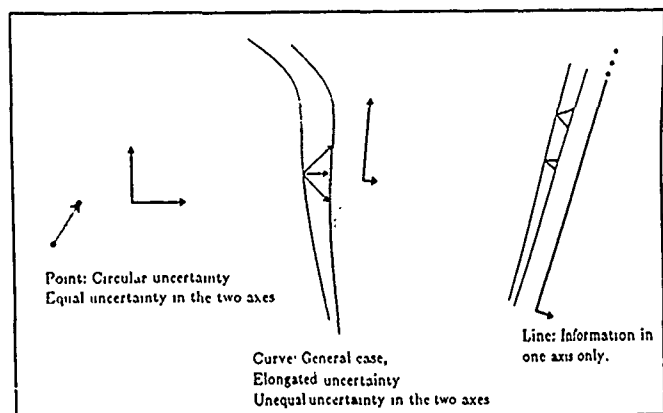


Figure 2. A curve is a general notion and a line or a point are special cases. Since the image plane is fully covered by isophotes practically any pixel can be used.

intuitively true that the uncertainty should decrease with this iteration. But a rigorous treatment of non-Gaussian noise is among our future plans. Summarizing, what we effectively do is postpone any attempt to find the undeterminable components of the displacement field, thus eliminating the necessity of the smoothness assumption; instead we use the rigidity assumption.

2 Geometric Analysis

2.1 Camera Geometry

We use here the pinhole camera model with a 3-D coordinate system $OXYZ$ and an image plane parallel to the XY plane at $Z=1$, which sets the focal length equal to unity. An object point $P_1 = [X_1, Y_1, Z_1]^T$ is projected on the image plane on the point $p_1 = \left[\frac{X_1}{Z_1}, \frac{Y_1}{Z_1}, 1 \right]^T$ (Figure 3). Often in the literature point p_1 is represented as a 2-D vector $p_1 = \left[\frac{X_1}{Z_1}, \frac{Y_1}{Z_1} \right]^T$ which is the same thing. We prefer, though, to stick to the 3-D vectors for one more reason beyond uniform notation: one can normalize the image vector p_1 to unit length $p_1' = \frac{p_1}{\|p_1\|}$ and thus avoid having the length of the image vector arbitrarily distort the weights in the least squares. Furthermore, when an object point P_1 rotates to $R \cdot P_1$, then the corresponding image points are p_1' and $R \cdot p_1'$ if they are normalized to unity and p_1 and $\frac{R \cdot p_1}{\hat{Z}^T \cdot R \cdot p_1}$ if they are not (\hat{Z} is the unit vector along the Z axis). This normalization simplifies things considerably. Note, though, that we don't use spherical coordinates but just a notation convenient for mathematical manipulations. From now on, we use normalized image

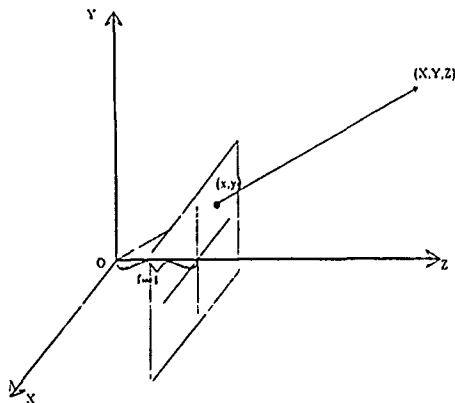


Figure 3. Basic camera geometry

point vectors.

2.2 Two Frames, Three Frames.

It has been proved [Tsai and Huang, 1984] that two frames, in the absence of noise, are enough to recover motion and structure. But if noise is present two frames are not enough! Even if we squeeze every bit of information out of the data by using an optimal algorithm [Spetsakis and Aloimonos, 1988b] we cannot do many things that we can do using more frames and it is easy to see why.

Suppose a point P_1 that belongs to a rigid object undergoes rigid motion with rotation matrix R_1 and translation vector T_1 and moves to P_2 . Then

$$P_2 = R_1 \cdot P_1 + T_1$$

or

$$\rho_2 \cdot p_2 = \rho_1 \cdot R_1 \cdot p_1 + T_1 \quad (1)$$

where ρ_1, ρ_2 are the lengths of P_1 and P_2 . Then the structure (e.g. the length of the vector P_1) is

$$\rho_1 = \frac{[(R_1 \cdot p_1) \times p_2]^2}{[(R_1 \cdot p_1) \times p_2]^T \cdot [T_1 \times p_2]}$$

If we consider a third frame then

$$P_3 = R_2 \cdot P_1 + T_2$$

or

$$\rho_3 \cdot p_3 = \rho_1 \cdot R_2 \cdot p_1 + T_2 \quad (2)$$

and the structure is now

$$\rho_1' = \frac{[(R_2 \cdot p_1) \times p_3]^2}{[(R_2 \cdot p_1) \times p_3]^T \cdot [T_2 \times p_3]}$$

In the absence of noise $\rho_1 = \rho_1'$ should hold. In the presence of noise it doesn't necessarily hold, if the motion parameters are computed using the frames pairwise. Forcing $\rho_1 = \rho_1'$ is one more constraint on the motion parameters. So we can get three equations for every point with three frames (Figure 4) instead of two if we use the frames pairwise. It is important to stress that this is not "yet another" equation. If we needed more equations we could use more points. But this extra equation changes things radically. To say the least the aperture problem is no longer a problem in this three-frame formulation, and we can easily expand to many frames, as we explain later.

With the need for three frames established we can go on and describe the theory. The only assumption we need is the rigidity assumption which is expressed by equations (1), (2) above. Indeed these equations are actually equivalent to the rigidity assumption because if a rigid body is undergoing a motion then a rotation matrix and a translation vector are enough to describe its motion in the fashion of equations (1), (2). If we now eliminate the structure unknowns ρ_1, ρ_2, ρ_3 from (1), (2) we are left with a matrix equation in terms of the data (the position of the image point p_1 in the

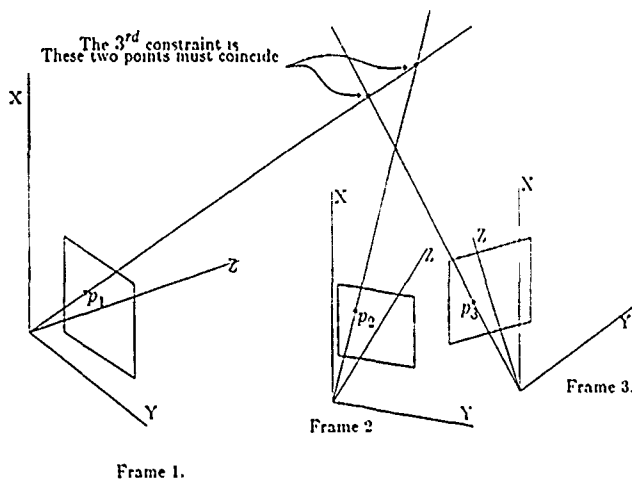


Figure 4. This third constraint is the critical one that makes the unification possible

three frames) of the form (see Appendix I for the derivation)

$$\tilde{p}_2 \cdot [(K, L, M) * p_1] \cdot \tilde{p}_3 = 0 \quad (3)$$

where if $p_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$ then

$$\tilde{p}_2 = \begin{bmatrix} 0 & z_2 & -y_2 \\ -z_2 & 0 & x_2 \\ y_2 & -x_2 & 0 \end{bmatrix}$$

and the same for p_3 . The matrices K, L, M are

$$\begin{aligned} K &= T_1 \cdot (R_2 \cdot \hat{X})^T - (R_1 \cdot \hat{X}) \cdot T_2^T \\ L &= T_1 \cdot (R_2 \cdot \hat{Y})^T - (R_1 \cdot \hat{Y}) \cdot T_2^T \\ M &= T_1 \cdot (R_2 \cdot \hat{Z})^T - (R_1 \cdot \hat{Z}) \cdot T_2^T \end{aligned} \quad (4)$$

and the operation $(K, L, M) * p_1$ is $x_1 \cdot K + y_1 \cdot L + z_1 \cdot M$ where $p_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$.

These three matrices do not appear here for the first time. In [Spetsakis and Aloimonos, 1987b] they appear in the solution of the line correspondence problem which here we also solve as part of the general problem.

Looking at expression (3) one can make some interesting observations. It is a matrix equation that is equivalent to nine scalar equations, only three of which are independent. They are non-linear in terms of the motion parameters but, since we replaced the non-linear terms with a set of 27 essential parameters (the elements of the three matrices K, L, M), are linear with respect to the essential parameters. For

every point we get three independent equations and we need at least nine points to find the essential parameters by solving the system of linear equations that come from applying (3) to several points.

2.3 Lines, Aperture Problem

Using equation (3), which is a general constraint, we can derive the constraints for several other problems. We show now how to solve the line correspondence problem and the aperture problem. Both results are very interesting: the line correspondence result because we derive the same solution as [Spetsakis and Aloimonos, 1987b] using a general theory, and the aperture result because for the first time a "linear" solution is presented (an optimal one is presented later in the paper) for the famous aperture problem.

Suppose an image point p_1 is moving and due to the aperture problem we do not know its position in the other two frames but only

$$p_2 = p'_2 + \beta_1 \cdot b_1 \quad \text{and} \quad p_3 = p'_3 + \gamma_1 \cdot c_1$$

where β_1, γ_1 are undetermined real numbers and b_1, c_1 are unit vectors parallel to the tangent directions. Since β_1, γ_1 are unknowns we eliminate them by pre and post-multiplying (3) by b_1, c_1 .

$$\begin{aligned} 0 &= b_1^T \cdot \tilde{p}_2 \cdot [(K, L, M) * p_1] \cdot \tilde{p}_3 \cdot c_1 = \\ &= (p_2 \times b_1)^T \cdot [(K, L, M) * p_1] \cdot (p_3 \times c_1) \end{aligned}$$

or

$$(p'_2 \times b_1)^T \cdot [(K, L, M) * p_1] \cdot (p'_3 \times c_1) = 0 \quad (5)$$

Now we have one equation with unknowns the elements of K, L, M , and using at least 26 points with tangential motion uncertainty, we can find the motion. So as a byproduct we have a "linear" algorithm that does not suffer from the aperture problem.

The line correspondence problem is essentially the same, if we represent a line ϵ_i as a vector normal to both the line and to a vector of an image point that belongs to the line. Let $\epsilon_1 = p'_1 \times a_1$, $\epsilon_2 = p'_2 \times b_1$, $\epsilon_3 = p'_3 \times c_3$ where p'_1 and a_1 are (as before) a point on the line and its direction, respectively. Then (5) implies

$$\epsilon_2^T \cdot [(K, L, M) * p_1] \cdot \epsilon_3 = 0$$

which holds for every image point $p_1 = p'_1 + \alpha_1 \cdot a_1$ where α_1 is any number. Then

$$\begin{bmatrix} \epsilon_2^T \cdot K \cdot \epsilon_3 \\ \epsilon_2^T \cdot L \cdot \epsilon_3 \\ \epsilon_2^T \cdot M \cdot \epsilon_3 \end{bmatrix} \cdot (p'_1 + \alpha_1 \cdot a_1) = 0$$

or

$$\begin{bmatrix} \epsilon_2^T \cdot K \cdot \epsilon_3 \\ \epsilon_2^T \cdot L \cdot \epsilon_3 \\ \epsilon_2^T \cdot M \cdot \epsilon_3 \end{bmatrix} \times \epsilon_1 = 0 \quad (6)$$

This is a vector equation that is equivalent to three scalar ones only two of which are independent [Liu et al., 1987; Spetsakis and Aloimonos, 1987b].

2.4 Linear Estimation of K, L, M

As we mentioned above, equation (3) is linear with respect to the essential parameters. We arrange the elements of these three matrices in a 27×1 vector x which is the column vector of the unknowns. Let A_i be the 9×27 matrix of coefficients of the unknowns in (3) for the i^{th} point. Then

$$x^T \cdot \left(\sum_i A_i^T A_i \right) \cdot x = 0$$

or

$$x^T \cdot A \cdot x = 0 \quad (7)$$

with condition $\|x\| = 1$, is the equation that gives vector x (the essential parameters) in the absence of noise. In the presence of noise, though, the r.h.s. of (7) is not going to be zero. In any case the solution for x is the eigenvector of A with the smallest eigenvalue (zero, if there is one).

This solution, although simple and nice, is not the best in the presence of noise. Indeed we have to minimize the l.h.s. of (7) but x is not just any vector. Its elements are the elements of the matrices K, L, M which are matrices with specific properties—e.g., all are singular, etc. Thus x belongs in a constraint space whereas the aforementioned eigenvector is just the solution to the unconstrained minimization. Although this solution is not the best one, it is a very good guess for the iterative procedure of constrained minimization that we describe later in the paper.

3 Deciphering the Essential Parameters

In the previous paragraphs we described a method to compute the matrices K, L, M using point or line matches or normal components of the flow. Assuming now that we know the matrices we shall describe how we can find the motion parameters from the matrices K, L, M .

These three matrices can be written as

$$K = \begin{bmatrix} T_1 & R_1 \hat{z} \end{bmatrix} \cdot \begin{bmatrix} (R_2 \hat{z})^T \\ -T_2^T \end{bmatrix}$$

$$L = \begin{bmatrix} T_1 & R_1 \hat{y} \end{bmatrix} \cdot \begin{bmatrix} (R_2 \hat{y})^T \\ -T_2^T \end{bmatrix}$$

$$M = \begin{bmatrix} T_1 & R_1 \hat{x} \end{bmatrix} \cdot \begin{bmatrix} (R_2 \hat{x})^T \\ -T_2^T \end{bmatrix}$$

Since they are products of two matrices of rank 2 they

are of rank at most 2 and hence at least one of their singular values is zero. Assuming for a moment that only one of them is zero we can easily see that the corresponding singular vectors are orthogonal to T_2 and to $R_2 \hat{z}$ for K , to $R_2 \hat{y}$ for L and to $R_2 \hat{x}$ for M . Since these three singular vectors cannot be collinear (due to their orthogonality to the columns of the matrix R_2) it is obvious that we can determine T_2 . But since up to two out of three matrices might have a second singular value there is a set of rare cases in which this simple method will not work. This set of cases is extremely rare when there is no noise but when noise is present then the solution is unstable if we are in the neighborhood of such cases.

In Appendix II we describe how to construct a 3×3 matrix J one eigenvalue of which is zero (in the absence of noise) and the corresponding eigenvector is parallel to T_2 . In the same way we can find the direction of T_1 . We now have to find the lengths of these vectors. Let f_1 be a unit vector normal to T_2 . Then

$$K \cdot f_1 = \begin{bmatrix} T_1 & R_1 \hat{z} \end{bmatrix} \cdot \begin{bmatrix} (R_2 \hat{z})^T \\ -T_2^T \end{bmatrix} \cdot f_1 =$$

$$\begin{bmatrix} T_1 & R_1 \hat{z} \end{bmatrix} \cdot \begin{bmatrix} \hat{z}^T \cdot R_2^T \cdot f_1 \\ 0 \end{bmatrix} = T_1 (\hat{z}^T \cdot R_2^T \cdot f_1)$$

Similarly

$$L \cdot f_1 = T_1 (\hat{y}^T \cdot R_2^T \cdot f_1)$$

$$M \cdot f_1 = T_1 (\hat{x}^T \cdot R_2^T \cdot f_1)$$

So

$$(K \cdot f_1)^2 + (L \cdot f_1)^2 + (M \cdot f_1)^2 =$$

$$= T_1^2 \cdot \left[(\hat{z}^T \cdot R_2^T \cdot f_1)^2 + (\hat{y}^T \cdot R_2^T \cdot f_1)^2 + (\hat{x}^T \cdot R_2^T \cdot f_1)^2 \right] = T_1^2 \neq 0$$

In brief, the procedure to find the translation vectors up to a sign is: Construct matrix J as described in Appendix II. Find the eigenvector with the smallest eigenvalue. This is the direction of T_2 . Take two unit vectors f_1 and f_n mutually orthogonal and both orthogonal to T_2 (the other two eigenvectors of J are OK). Then

$$T_1^2 = \frac{1}{2} \left[(K \cdot f_1)^2 + (L \cdot f_1)^2 + (M \cdot f_1)^2 + \right.$$

$$\left. + (K \cdot f_n)^2 + (L \cdot f_n)^2 + (M \cdot f_n)^2 \right]$$

Exactly the same is done for the direction of T_1 and the length of T_2 .

Now we describe the procedure to find the rotation parameters. Let f_1, f_2, f_3 be three non-collinear unit vectors that are all orthogonal to T_2 . As we have already seen, for vectors normal to T_2

$$K \cdot f_1 = T_1 (\hat{z}^T \cdot R_2^T \cdot f_1), L \cdot f_1 = T_1 (\hat{y}^T \cdot R_2^T \cdot f_1),$$

$$M \cdot f_1 = T_1 \cdot (\hat{z}^T \cdot R_2^T \cdot f_1)$$

and in the same way

$$K \cdot f_2 = T_1 \cdot (\hat{x}^T \cdot R_2^T \cdot f_2), L \cdot f_2 = T_1 \cdot (\hat{y}^T \cdot R_2^T \cdot f_2),$$

$$M \cdot f_2 = T_1 \cdot (\hat{z}^T \cdot R_2^T \cdot f_2)$$

$$K \cdot f_3 = T_1 \cdot (\hat{x}^T \cdot R_2^T \cdot f_3), L \cdot f_3 = T_1 \cdot (\hat{y}^T \cdot R_2^T \cdot f_3),$$

$$M \cdot f_3 = T_1 \cdot (\hat{z}^T \cdot R_2^T \cdot f_3)$$

Define

$$F_2 = \begin{bmatrix} f_1 & f_2 & f_3 \end{bmatrix}$$

to be a 3×3 matrix whose columns are f_1, f_2, f_3 . Then we can compute up to a sign a matrix A_2 such that

$$A_2 = s_2 \cdot R_2^T \cdot F_2$$

The s_2 represents the sign ambiguity that was inherited from the computation of the translation vector; it is an unknown.

The SVD's (Singular Value Decompositions) of A_2, F_2 are

$$A_2 = U_2 \cdot \Sigma_2 \cdot V_2^T \text{ and } F_2 = X_2 \cdot \Sigma_2 \cdot V_2^T$$

These are a set of the possibly infinite number of different SVD's these two matrices can be decomposed into. We can always find a set that shares the matrices V_2 and Σ_2 because [Stewart, 1973; Strang, 1980]

$$A_2^T \cdot A_2 = F_2^T \cdot F_2 = V_2 \cdot \Sigma_2^2 \cdot V_2^T$$

Using a standard routine to find the SVD's for both of them does not guarantee that they will share V_2 and Σ_2 . To achieve this we can compute the SVD of A_2 , from which we know V_2 and Σ_2 , and from

$$X_2 \cdot \Sigma_2 = F_2 \cdot V_2$$

we can find

$$X_2 = \begin{bmatrix} \pm X_{21} & X_{22} & X_{23} \end{bmatrix}$$

The ambiguity in the sign comes from the zero singular value. There is only one singular value because the three f_i 's are not collinear.

So

$$s_2 A_2 = r_2^T \cdot F_2$$

which implies

$$s_2 U_2 \cdot \Sigma_2 \cdot V_2^T = R_2^T \cdot X_2 \cdot \Sigma_2 \cdot V_2^T$$

By premultiplying and postmultiplying by U_2^T and V_2 respectively we get

$$s_2 \Sigma_2 = U_2^T \cdot R_2^T \cdot X_2 \cdot \Sigma$$

Then obviously

$$U_2^T \cdot R_2^T \cdot X_2 = \begin{bmatrix} \sigma_2 & & \\ & s_2 & \\ & & s_2 \end{bmatrix}$$

where $\sigma_2 = \det(U_2) \cdot \det(X_2)$. The σ_2 takes values ± 1 and this enforces the R_2 to be right handed. Finally

$$R_2 = X_2 \cdot \begin{bmatrix} \sigma_2 & & \\ & s_2 & \\ & & s_2 \end{bmatrix} \cdot U_2^T \quad (8)$$

In a similar way

$$R_1 = X_1 \cdot \begin{bmatrix} \sigma_1 & & \\ & s_1 & \\ & & s_1 \end{bmatrix} \cdot U_1^T \quad (9)$$

There are four possible combinations of signs s_1, s_2 . This ambiguity is due to the lack of information along the direction of T_1 and T_2 . In [Spetsakis and Aloimonos, 1987a] it is explained how one can resolve this ambiguity.

4 Statistical Analysis

So far noise was treated as a secondary issue, with geometric constraints being the main focus. But with all the necessary theoretical tools involving the geometry of the problem at hand we can deal with the noise at the level it deserves. Working in the framework described in [Spetsakis and Aloimonos, 1988b] we derive the expression to be minimized and develop the procedure to do so.

The vector equations (1), (2) are no longer adequate. We must add a term to stand for the noise. We choose to introduce a 3-D noise vector; although it does not seem to have a unique meaning, it simplifies the equations, and in the final expression only a function of it appears which is equal to the image plane noise, and this is precisely what we need.

We rewrite equations (1), (2) to reflect the presence of noise:

$$\rho_1 \cdot p_2 = \rho_1 \cdot R_1 \cdot p_1 + T_1 + n_2$$

$$\rho_3 \cdot p_3 = \rho_1 \cdot R_2 \cdot p_1 + T_2 + n_3$$

With some complex mathematical manipulations, described in Appendix III, we get finally

$$\tilde{p}_2 \cdot \left[(K, L, M) * p_1 \right] \cdot \tilde{p}_3 -$$

$$= \frac{-1}{\rho_1} \cdot \left[(p_2 \times T_1) \cdot (p_3 \times n_3)^T - (p_2 \times n_2) \cdot (p_3 \times T_2)^T \right] = Q$$

Vector $p_3 \times \frac{n_3}{\rho_1}$ is always normal to ρ_3 and contains the visible component of the 3-D noise vector divided

by ρ_1 so it is very close to being of equal length to the image noise vector. In order for it to be equal to the image noise vector it should be divided by ρ_3 , but we show how this can be taken care of later in this paper. We can analyze it in two components such that the two of them are independent (something we can always do if the noise is Gaussian). So $p_3 \times \frac{n_3}{\rho_1} = \gamma_1 c_1 + \gamma_2 c_2$. In the same fashion we analyze $p_2 \times \frac{n_2}{\rho_1} = \beta_1 b_1 + \beta_2 b_2$. Of course b_1, b_2, p_2 form an orthonormal system and the so do c_1, c_2, p_3 . Thus

$$Q = -(p_2 \times T_1) \cdot (\gamma_1 c_1 + \gamma_2 c_2)^T + (\beta_1 b_1 + \beta_2 b_2) \cdot (p_3 \times T_2)^T$$

In the previous paragraphs we explained why this is equivalent to three equations. Any choice of three equations though will have instances in which two of them are dependent, causing abnormally high numbers to appear in the inverse of the covariance matrix. So we take four of them and we pick them so that they are symmetric:

$$b_1^T \cdot Q \cdot c_1 = \gamma_1 (b_2^T \cdot T_1) - \beta_1 (c_2^T \cdot T_2)$$

$$b_1^T \cdot Q \cdot c_2 = \gamma_2 (b_2^T \cdot T_1) + \beta_1 (c_1^T \cdot T_2)$$

$$b_2^T \cdot Q \cdot c_1 = -\gamma_1 (b_1^T \cdot T_1) - \beta_2 (c_2^T \cdot T_2)$$

$$b_2^T \cdot Q \cdot c_2 = -\gamma_2 (b_1^T \cdot T_1) + \beta_2 (c_1^T \cdot T_2)$$

We can write this in matrix form after substituting $b_1^T \cdot T_1 = v_1, b_2^T \cdot T_1 = v_2, c_1^T \cdot T_2 = g_1, c_2^T \cdot T_2 = g_2$:

$$W = \Omega \cdot N$$

where

$$W = \begin{bmatrix} b_1^T \cdot Q \cdot c_1 \\ b_1^T \cdot Q \cdot c_2 \\ b_2^T \cdot Q \cdot c_1 \\ b_2^T \cdot Q \cdot c_2 \end{bmatrix}, \quad N = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \gamma_1 \\ \gamma_2 \end{bmatrix}$$

and

$$\Omega = \begin{bmatrix} -g_2 & 0 & v_2 & 0 \\ g_1 & 0 & 0 & v_2 \\ 0 & -g_2 & -v_1 & 0 \\ 0 & g_1 & 0 & -v_1 \end{bmatrix}$$

The covariance matrix of W is

$$E[W \cdot W^T] = \Omega \cdot E[N \cdot N^T] \cdot \Omega^T \quad (10)$$

In weighted least squares estimation we need the inverse of the covariance matrix. But the one given by (10) is non-invertible because it represents three equations while it is 4×4 . This means no more than that the probability for the vector W to be in the left null-space of Ω (e. g. to have non-trivial component there) is exactly 0. So since vector W will not have any component in this null space we can set the

variance in this space arbitrarily. The Hawking-Penrose pseudo-inverse then is a good choice since it sets this variance to zero. The standard procedure to find the pseudo-inverse of a matrix is to take its SVD, invert the non-zero singular values, leave the zero ones zero, interchange the left and right orthonormal matrices, and multiply. Since the zero singular values correspond to improbable events the arbitrary choice of zero instead of their inverse does not affect the end result.

The pseudo-inverse of $E[W \cdot W^T]$ is

$$E[W \cdot W^T]^+ = \Omega^+ \cdot E[N \cdot N^T]^{-1} \cdot \Omega^+ \quad (11)$$

In Appendix IV we deal with the analytic computation of Ω^+ . The most interesting thing is the singular values of Ω^+ . The diagonal matrix of the singular values is

$$\Sigma^+ = \begin{bmatrix} \frac{1}{\sqrt{v_1^2 + v_2^2}} & & & \\ & \frac{1}{\sqrt{g_1^2 + g_2^2}} & & \\ & & \frac{1}{\sqrt{g_1^2 + g_2^2 + v_1^2 + v_2^2}} & \\ & & & 0 \end{bmatrix}$$

What makes the above matrix interesting is the range of the diagonal elements:

$$v_1^2 + v_2^2 = (b_1^T \cdot T_1)^2 + (b_2^T \cdot T_1)^2 = T_1^2 - (p_2 \cdot T_1)^2 = (T_1 \times p_2)^2$$

Since $(T_1 \times p_2)^2$ can take then values from the interval $[0, \|T_1\|^2]$, the singular value $\frac{1}{\sqrt{v_1^2 + v_2^2}}$ can

take values from $(\frac{1}{\|T_1\|}, \infty)$. These singular values

represent the weight of the least squares, and from the above expression it is obvious that solutions for the translation that are close to an image point vector get very high weight. There is a reason for this: if they did not get this weight then in the presence of noise we would see a tendency for the translations to be closer to the center of gravity of the image points. If for example the actual translation vectors form an angle θ with the center of gravity vector of the image points, then the computed translation vectors will form angles, yielding a mean noticeably less than θ . So the singular values are there to penalize solutions for translation that tend to be biased towards the center of gravity. This phenomenon is quite interesting but nonetheless not new. Exactly the same situation was encountered in [Spetsakis and Aloimonos, 1987b] although there there were only two frames and one equation per point so there were no singular values; the weight was a real number and equal to $[T_1 \times p_2]^2$ if we use the same conventions.

Finally as we mentioned earlier $\|p_3 \times \frac{n_3}{\rho_1}\|$ does not represent the image noise which actually is $\|p_3 \times \frac{n_3}{\rho_3}\|$. This aberration is passed down to γ_1 ,

γ_2 , and the same for β_1, β_2 . We just have to multiply β_1, β_2 by $\frac{\rho_1}{\rho_2}$ and γ_1, γ_2 by $\frac{\rho_1}{\rho_3}$, which are approximately, from the law of the sines:

$$\frac{\rho_1}{\rho_2} = \frac{\|T_1 \times p_2\|}{\|T_1 \times p'_1\|}, \quad \frac{\rho_1}{\rho_3} = \frac{\|T_2 \times p_3\|}{\|T_2 \times p''_1\|}$$

where $p'_1 = R_1 \cdot p_1$ and $p''_1 = R_2 \cdot p_1$. Now we can write

$$E[N \cdot N^T] = \begin{bmatrix} E[\beta_1^2] & & & \\ & E[\beta_2^2] & & \\ & & E[\gamma_1^2] & \\ & & & E[\gamma_2^2] \end{bmatrix} = \begin{bmatrix} E[\beta_1^2] \frac{\rho_1}{\rho_2} & & & \\ & E[\beta_2^2] \frac{\rho_1}{\rho_2} & & \\ & & E[\gamma_1^2] \frac{\rho_1}{\rho_3} & \\ & & & E[\gamma_2^2] \frac{\rho_1}{\rho_3} \end{bmatrix}$$

where $\beta_1 = \beta'_1 \frac{\rho_1}{\rho_2}$ etc. The variances $E[\beta_1^2] \dots E[\gamma_1^2]$ are supposed known because they represent the uncertainty in the measurements of the displacement vectors.

5 Minimization Process

We have now to solve the problem of doing the non-linear minimization which we break into three successive minimization steps, the first two approximate to get a good initial guess, and the last one that converges to the minimum.

As we mentioned above vector W should have

value $W = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ but due to noise it is of non-trivial

value. So we do weighted least squares (as we see from a simple maximum likelihood argument) on $\sum_i W_i^T \cdot C_i \cdot W_i$, where W_i is the expression W for the i^{th} point and $C_i = E[W_i \cdot W_i^T]^{-1}$, which we showed how to compute analytically in the previous section. This expression can be minimized using any commercial package for optimization but would prove very expensive in computation time if a random guess is used. Instead we present a method to derive a good suboptimal solution which if fed into a general purpose routine serves as a good guess and makes it converge

to the optimal solution much faster. The process is not totally different from what is done in the case of two frames [Spetsakis and Aloimonos, 1988b]. We first get a "linearized" solution using unconstrained minimization, then use a constrained minimization scheme to improve the linearized solution which is now close enough to the optimal so that a general routine can be used efficiently.

Let x be a 27×1 vector which consists of the elements of the matrices K, L, M one on top of the other. Define A_i to be the matrix of the coefficients of the elements of the three matrices that give the vector W_i , e. g.

$$W_i = A_i \cdot x$$

Of course A_i is 4×27 (remember that the equations are linear with respect to the elements of the matrices). Then set C_i equal to some matrix C'_i , independent of the motion parameters. If there is a reasonable guess for the motion parameters then one can use this to construct C'_i from these and if there is no guess one can set C'_i to be a diagonal matrix whose diagonal elements are the sum of the squares of the noise variables involved in each entry of the vector W_i .

After setting a value for C'_i , the expression to be minimized is

$$x^T \cdot \left[\sum_i A_i^T \cdot C'_i \cdot A_i \right] \cdot x$$

or

$$x^T \cdot A \cdot x$$

with $\|x\|$ constant.

The eigenvector of A with the smallest eigenvalue is the solution for x which minimizes the above quadratic. This vector x is supposed to be the elements of matrices K, L, M that are decomposable to motion parameters. But it is not. So we find motion parameters that give matrices K, L, M close to the ones that come from x . The process of finding the motion parameters was described in the previous section. The second step is to take into account that x is not just any vector but one that belongs to a subspace which is non-linear. All the vectors x that are decomposable to motion parameters belong in this space. If we define the vector ζ to be $\zeta = [b_1, b_2, \dots, b_6, t_1 t_2, \dots, t_6]^T$, where b_1, b_2, \dots, b_6 are the Rodrigues parameters for the matrices R_1, R_2 and $t_1 t_2, \dots, t_6$ are the components of the translation vectors T_1, T_2 , then x has the Taylor expansion

$$x(\zeta + \Delta\zeta) = x(\zeta) + D(\zeta) \cdot \Delta\zeta + \text{higher order terms}$$

The column space of $D(\zeta)$ is a linear space tangent to the non-linear space where x belongs. D is a 27×12 matrix of the derivatives of x with respect to ζ . Now one can minimize the quadratic $x^T \cdot A \cdot x$ in the column space of D . The solution again is not a decomposable vector x so we find one close by that belongs. This is a new guess, improved, and can be used for a new

iteration of finding D and then minimizing. This is expected to converge quickly.

Now the solution for z is a very good guess that can be used for an iterative algorithm to find the optimal solution. The complicated method we used to get a good guess is justified by the high cost of computing matrix A (to do this we have to go through all points, now these are on the order of the number of pixels since with the unified definition practically all of them can be used). Matrix A has to be computed at every iteration of a general purpose algorithm whereas for the approximate ones it doesn't need to be computed more than once.

6 Many Frames

The main issues in the problem of computing structure given optic flow, correspondence, or anything equivalent are instability and lack of uniformity in representation. Almost all the well-known algorithms are sensitive to random perturbation of their input and not compatible with each other. In this paper, so far, we have presented very optimistic results on the issue of uniformity in representation by unifying the definition of "features" in such a way that their motion can be described with the same set of parameters. In this way we were able to design an algorithm that deals with all of them in a unified fashion. We derived the expression that if minimized yields the optimal solution for three frames. Thus we achieved two steps towards noise stability at the same time: we increased the number and the type of features that can be used in the motion estimation process (in this framework the whole image can be used), and we utilized all the information contained in them. But feature redundancy and optimality are not enough. One thing any biological vision system has, and which, so far, is not properly used in artificial systems, is redundancy in the frames. We need to be able to incorporate many frames without severe increase in computation time. In our previous work [Spetsakis and Aloimonos, 1988a] such a solution is presented, but it cannot fit into the statistical paradigm of the present approach (in other words, it is restricted to deal only with points), and its complexity is $O(N^4)$ for each iteration, where N is the number of frames used. A truly real time system should be able to perform in time linear in the number of frames.

Given three frames it is easy to extend to more. The basic idea is recursion. Assume the system has three frames in its memory. It computes the motion as we described, then computes the structure, as we describe below (Figure 5). Now the system has the best estimate of the structure in its memory. Now a new frame comes in from the camera. How can the system update its estimate of the structure? It is well known that two frames are enough, at least in principle, to recover structure and motion. Hence the opposite is also true: if we have the shape of an object we can find what its image would be in two different projections. So the system can encode the structure

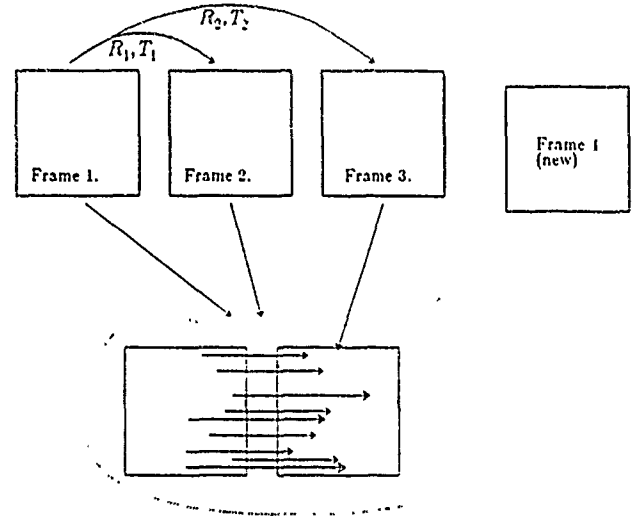


Figure 5. The three frames can be encoded as the displacement of two pairs. The uncertainty in this displacement is less than initially. The two, along with a fourth one, are again a set of three frames.

information, that was computed from the three frames, in two frames that have less uncertainty than the original three. Now the two frames that encode the structure plus the fresh one can be used to derive an even better estimate of structure.

6.1 Computing Structure

The structure is not hard to compute. Taking into consideration the noise, equations (1), (2) are

$$\rho_2(p_2 + \beta_1 b_1 + \beta_2 b_2) = \rho_1(R_1 p_1) + T_1$$

$$\rho_3(p_3 + \gamma_1 c_1 + \gamma_2 c_2) = \rho_1(R_2 p_1) + T_2$$

$$\text{Set } \beta^*_1 = \rho_2 \beta_1, \beta^*_2 = \rho_2 \beta_2, \gamma^*_1 = \rho_3 \gamma_1, \gamma^*_2 = \rho_3 \gamma_2.$$

$$\rho_2 p_2 + \beta^*_1 b_1 + \beta^*_2 b_2 = \rho_1 p'_1 + T_1$$

$$\rho_3 p_3 + \gamma^*_1 c_1 + \gamma^*_2 c_2 = \rho_1 p''_1 + T_2$$

Then the solution of the system

$$A \cdot x = b \quad (12)$$

$$x^T \cdot C \cdot x = \min \quad (13)$$

gives the structure for every point, where

$$A = \begin{bmatrix} x_2 & b_{x1} & b_{x2} & -x'_1 & 0 & 0 & 0 \\ y_2 & b_{y1} & b_{y2} & -y'_1 & 0 & 0 & 0 \\ z_2 & b_{z1} & b_{z2} & -z'_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -x''_1 & x_3 & c_{x1} & c_{x2} \\ 0 & 0 & 0 & -y''_1 & y_3 & c_{y1} & c_{y2} \\ 0 & 0 & 0 & -z''_1 & z_3 & c_{z1} & c_{z2} \end{bmatrix}$$

$$b = \begin{bmatrix} t_{z1} \\ t_{y1} \\ t_{x1} \\ t_{z2} \\ t_{y2} \\ t_{x2} \end{bmatrix}, \quad x = \begin{bmatrix} \rho_2 \\ \beta^*_{x1} \\ \beta^*_{x2} \\ \rho_1 \\ \rho_3 \\ \gamma^*_{x1} \\ \gamma^*_{x2} \end{bmatrix}$$

and

$$C = \begin{bmatrix} 0 & & & & & & \\ & \frac{E[\beta_1^2]^{-1}}{\rho_2^2} & & & & & \\ & & \frac{E[\beta_2^2]^{-1}}{\rho_2^2} & & & & \\ & & & 0 & & & \\ & & & & 0 & & \\ & & & & & \frac{E[\gamma_1^2]^{-1}}{\rho_2^2} & \\ & & & & & & \frac{E[\gamma_2^2]^{-1}}{\rho_2^2} \end{bmatrix}$$

From equation (12) we get, using the Hawking-Penrose pseudo-inverse,

$$x_H = A^+ \cdot b$$

We try to estimate the parameter α so that the solution $x = x_H + \alpha x_N$ where x_N belongs in the null space of A , minimizes (13). We can either minimize (13) accurately (it ends up to being the solution of a seventh degree polynomial) or use approximate values for ρ_2, ρ_3 and consider C , which depends on them, to be constant:

$$\rho_2 = \frac{\|T_1 \times p'_1\|}{\|p'_1 \times p_2\|}, \quad \rho_3 = \frac{\|T_2 \times p''_1\|}{\|p''_1 \times p_3\|}$$

Then (13) becomes

$$(x_H + \alpha x_N)^T \cdot C \cdot (x_H + \alpha x_N) =$$

$$x_H^T \cdot C \cdot x_H + \alpha (x_N^T \cdot C \cdot x_H + x_H^T \cdot C \cdot x_N) + \alpha^2 x_N^T \cdot C \cdot x_N$$

and the minimum occurs for

$$\alpha = \frac{-x_N^T \cdot C \cdot x_H}{x_N^T \cdot C \cdot x_N}$$

This computation is very simple and the better estimates for ρ_2, ρ_3 that we get from the α we estimated

can be used in another iteration to get a new C and then a new α . This phenomenon, where the computation of the structure that minimizes the squared error is the solution of some non-linear system of equations, appears in two-frame optimal estimation [Weng et al., 1987]. The intuition behind the two is the same. If it was the distance of the 3-D point from the extensions of the image point vectors that we wanted to minimize then the equations would be linear. But since we want to minimize the projections of these distances we have to use non-linear equations. The two solutions are almost the same except that when minimizing for the projections the solution tends to be a little farther away so that the projections become smaller.

6.2 Multiple Frame Iteration

Now that we know the structure we have to find how much we can trust what we have found, in other words find the uncertainty. It was the knowledge of the uncertainty that permitted us to unify a broad class of motion problems. Again the uncertainty will help us transfer information from one set of frames to the other.

Let $m = x^T \cdot C \cdot x$. Vector x can have only one degree of freedom and we choose this to be ρ_1 instead of α . Then the variance of the estimate is approximately

$$\frac{1}{\text{var} \rho_1} = E \left[\frac{\partial^2 m}{\partial \rho_1^2} \right]$$

as can be verified from any textbook on statistics.

So $\text{var} \rho_1 = \frac{\rho_{1N}^2}{2 x_N^T \cdot C \cdot x_N}$ where ρ_{1N} is the fourth component of the vector x_N . Introducing an artificial translation vector T_o the induced displacement vector for an image point vector p_1 (with the convention that the image point vectors are unitary) is

$$\frac{p_1 \times (p_1 \times T_o)}{\rho_1}$$

which again approximately has an uncertainty along its direction which is (written as a product of the unit vector along its direction and the variance)

$$\frac{p_1 \times (p_1 \times T_o)}{2 x_N^T \cdot C \cdot x_N} \cdot \frac{\rho_{1N}^2}{\rho_1^2}$$

It seems that we have found the uncertainty for the artificial motion but we are still missing something. The reason we have used this artificial translation T_o , which we can choose arbitrarily, is to encode the structure we computed from the three frames. There are two kinds of error introduced in the structure computation. one kind is due to uncertainty in correspondence and is uncorrelated for all the points for which we computed the structure, and the other kind is due to the uncertainty in estimating the motion parameters. The latter introduces a highly correlated error which deforms the 3-D scene smoothly. In computing (above) the artificial disparity vector and its

uncertainty for every point, we took into account only the first kind of error. So we had uncertainty in one direction only. The two uncertainties will be in orthogonal directions so a good choice is to pick the uncertainty in the second direction as the same as the first.

Interesting questions for future research are to find which choice of T_0 and uncertainties leads to the fastest convergence of the structure computation, and also the possibility of designing an optimal algorithm for many frames that does not have to store all the previous data but only a small number of them.

The importance of this algorithm is that it incorporates many frames with minimal increase of computation time and storage. Although it is based on an optimal unified algorithm, it doesn't so far have the advantage of proven optimality, but it is still a unified algorithm.

APPENDIX I

Equations (1),(2)

$$\rho_2 p_2 = \rho_1 R_1 \cdot p_1 + T_1$$

$$\rho_3 p_3 = \rho_1 R_2 \cdot p_1 + T_2$$

are two vector equations that represent the rigidity condition. They are equivalent to six scalar equations. We want to eliminate ρ_1 , ρ_2 , and ρ_3 . To do this we first take the cross product of the equations with p_2 and p_3

$$0 = \rho_1 p_2 \times (R_1 \cdot p_1) + p_2 \times T_1$$

$$0 = \rho_1 p_3 \times (R_2 \cdot p_1) + p_3 \times T_2$$

We rearrange the terms

$$\rho_1 p_2 \times (R_1 \cdot p_1) = -p_2 \times T_1$$

$$p_3 \times T_2 = -\rho_1 p_3 \times (R_2 \cdot p_1)$$

and then take the outer product of both sides

$$\tilde{p}_2 (R_1 \cdot p_1 \cdot T_2^T) \cdot \tilde{p}_3 = \tilde{p}_2 \cdot T_1 \cdot (R_2 \cdot p_1)^T \cdot \tilde{P}_3$$

where

$$\tilde{p}_2 = \begin{bmatrix} 0 & z_2 & -y_2 \\ -z_2 & 0 & x_2 \\ y_2 & -x_2 & 0 \end{bmatrix}$$

and

$$p_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$$

and the same for \tilde{p}_3 .

$$\text{If } p_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \text{ then}$$

$$\tilde{p}_2 \cdot (x_1 \cdot (R_1 \cdot \hat{x}) + y_1 \cdot (R_1 \cdot \hat{y}) + z_1 \cdot (R_1 \cdot \hat{z})) \cdot T_2^T \cdot \tilde{p}_3 =$$

$$\tilde{p}_2 \cdot T_1 \cdot (x_1 \cdot (R_2 \cdot \hat{x})^T + y_1 \cdot (R_2 \cdot \hat{y})^T + z_1 \cdot (R_2 \cdot \hat{z})^T) \cdot \tilde{p}_3$$

By defining

$$K = T_1 \cdot (R_2 \cdot \hat{x})^T - (R_1 \cdot \hat{x}) \cdot T_2^T$$

$$L = T_1 \cdot (R_2 \cdot \hat{y})^T - (R_1 \cdot \hat{y}) \cdot T_2^T$$

$$M = T_1 \cdot (R_2 \cdot \hat{z})^T - (R_1 \cdot \hat{z}) \cdot T_2^T$$

we get

$$\tilde{p}_2 \cdot (x_1 \cdot K + y_1 \cdot L + z_1 \cdot M) \cdot \tilde{p}_3 = \begin{bmatrix} 0 \end{bmatrix}$$

and finally

$$\tilde{p}_2 \cdot [(K, L, M) * p_1] \cdot \tilde{p}_3 = \begin{bmatrix} 0 \end{bmatrix} \quad (\text{a.1})$$

with the obvious definition of $(\cdot, \cdot, \cdot) * \cdot$.

Now we have to show that eq. (a.1) is equivalent to a set of nine equations of which only three are independent. It is easy to see that $(K, L, M) * p_1$ is a singular matrix for every vector p_1 . The left singular vector of the zero singular value for this matrix is normal to p_2 . This is one condition. The other two come from the fact that $\tilde{p}_2 [(K, L, M) * p_1]$ has only one non-zero singular value the corresponding singular vector of which is parallel to p_3 . These are three constraints in all.

APPENDIX II

If K, L, M are not degenerate (have two eigenvalues equal) then each one has an eigenvector f_1, f_2, f_3 which corresponds to the zero eigenvalue and is orthogonal to T_2 . If we know f_1, f_2, f_3 then T_2 is a left eigenvector of

$$F_2 = \begin{bmatrix} f_1 & f_2 & f_3 \end{bmatrix}$$

F_2 is a 3×3 matrix whose columns are f_1, f_2, f_3 . This eigenvector corresponds to the smallest eigenvalue, which means it is closest to being normal to vectors f_1, f_2, f_3 . But the solution cannot be complete because K, L, M can be degenerate or almost degenerate so that the above method cannot always work. If one tries to identify the problematic cases and treat them separately, then one is going to run into difficulties in the presence of noise where the border between degenerate and non-degenerate is vague.

In order to avoid this we make the following observations:

- Matrices K, L, M are equal to $(K, L, M) * p$ for $p = \hat{x}, \hat{y}, \hat{z}$.
- The matrix $(K, L, M) * p$ where p is a unit vector is non-degenerate for every p , with the exception of a set of measure zero, because

$$(K, L, M) * p = T_1 \cdot (R_2 \cdot p)^T - (R_1 \cdot p)^T \cdot T_2$$

and in order to be degenerate, either $p = \alpha R_1^T \cdot T_1$ or $p = \alpha R_1^T \cdot T_1$ for some real α .

The adjoint of $(K, L, M) * p$, $\text{adj}((K, L, M) * p) = N(p)$, is a matrix with two zero eigenvalues, and the eigenvector of the third eigenvalue (which is the product of the two non-zero eigenvalues of $(K, L, M) * p$) is the same as the eigenvector with zero eigenvalue of $(K, L, M) * p$. So the columns of $N(p)$ are parallel to this eigenvector. If c_1 is a column of $N(P)$ then

$$c_1^T \cdot T_2 = 0$$

This is one equation. Using the rest of the columns does not offer more constraints but in the presence of noise they contribute to the stability. The least squares expression for them is

$$T_2^T \cdot N(p) \cdot N(p)^T \cdot T_2 \rightarrow \min$$

The vector T_2 that minimizes the above must be normal to the eigenvector with zero eigenvalue of $(K, L, M) * p$. Notice that the above expression is implicitly weighted by the product of the two other eigenvalues of $(K, L, M) * p$. If one of them is zero or close to zero then their product is close too.

The above least squares expression represents one equation only, at least in the absence of noise. We need at least one more to have a solution. The method mentioned at the beginning of this appendix took this equation for $p = \hat{x}, \hat{y}, \hat{z}$, but had the risk that two of these p 's lead to degenerate matrices. One way around this is to find p 's that do not display this property. A better way is to use all of them by integrating with respect to p over the unit sphere. So we have to minimize

$$T_2^T \cdot J \cdot T_2$$

where

$$J = \int_{\text{unit sphere}} N(p) \cdot N(p)^T dp$$

If $p = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$ then

$$N(p) = x^2 \text{adj}(k) + x y \cdot KL + y^2 \text{adj}(L) + \\ + y z \cdot LM + z^2 \text{adj}(M) + z x \cdot MK$$

where

$$KL = \text{adj}(K + L) - \text{adj}(K) - \text{adj}(L)$$

$$LM = \text{adj}(L + M) - \text{adj}(L) - \text{adj}(M)$$

$$MK = \text{adj}(M + K) - \text{adj}(M) - \text{adj}(K)$$

The product $N(p) \cdot N(p)^T$ is a fourth degree homogeneous polynomial in x, y, z . To integrate each term of the polynomial separately we take into consideration

$$\int x^4 = \int y^4 = \int z^4 = 4 \frac{\pi}{5}$$

$$\int x^2 y^2 = \int y^2 z^2 = \int z^2 x^2 = \frac{4\pi}{15}$$

where the integration is done over the unit sphere. The integrals of all the other terms are zero.

So finally

$$J = 3(\text{adj}(K) \cdot \text{adj}(K)^T + \text{adj}(L) \cdot \text{adj}(L)^T + \text{adj}(M) \cdot \text{adj}(M)^T) +$$

$$\text{adj}(K) \cdot \text{adj}(L)^T + \text{adj}(L) \cdot \text{adj}(M)^T + \text{adj}(M) \cdot \text{adj}(K)^T +$$

$$\text{adj}(K) \cdot \text{adj}(M)^T + \text{adj}(M) \cdot \text{adj}(L)^T + \text{adj}(L) \cdot \text{adj}(K)^T +$$

$$KL \cdot KL^T + LM \cdot LM^T + MK \cdot MK^T +$$

This is a very easy procedure to compute T_2 . One has only to compute J , which is a series of adjoint matrix computations, matrix multiplications and additions, and then find its eigenvector with the smallest eigenvalue. The same, of course, for T_1 .

APPENDIX III

We rewrite equations (1),(2) to reflect the presence of noise:

$$\rho_2 p_2 = \rho_1 (R_1 \cdot p_1) + T_1 + n_2$$

$$\rho_3 p_3 = \rho_1 (R_2 \cdot p_1) + T_2 + n_3$$

By taking the left cross product with p_2 and p_3 and rearranging the terms we get

$$\rho_1 p_2 \times (R_1 \cdot p_1) = -p_2 \times T_1 - p_2 \times n_2$$

$$p_3 \times n_3 + p_3 \times T_2 = -\rho_1 p_3 \times (R_2 \cdot p_1)$$

We take the outer product of these two equations

$$\tilde{p}_2 \cdot (R_1 \cdot p_1 \cdot T_1^T) \cdot \tilde{p}_3 + \frac{1}{\rho_1} \tilde{p}_2 \cdot (R_1 \cdot p_1) \cdot n_3^T \cdot \tilde{p}_3 =$$

$$\tilde{p}_2 \cdot T_1 \cdot (R_2 \cdot p_1)^T \cdot \tilde{p}_3 + \frac{1}{\rho_1} \tilde{p}_2 \cdot n_2 \cdot (R_2 \cdot p_1) \cdot \tilde{p}_3$$

which implies

$$\tilde{p}_2 \cdot [(K, L, M) * p_1] \cdot \tilde{p}_3 =$$

$$= \frac{1}{\rho_1} [(p_2 \times p'_1) \cdot (p_3 \times n_3)^T - (p_2 \times n_2) \cdot (p_3 \times p''_1)^T]$$

APPENDIX IV

The matrix

$$\Omega = \begin{bmatrix} -g_2 & 0 & v_2 & 0 \\ g_1 & 0 & 0 & v_2 \\ 0 & -g_2 & -v_1 & 0 \\ 0 & g_1 & 0 & -v_1 \end{bmatrix}$$

has a singular value decomposition

$$\Omega = U \cdot \Sigma \cdot V^T$$

that can be computed analytically. Then the pseudoinverse can be computed analytically too.

Let $G = \sqrt{g_1^2 + g_2^2}$, $V = \sqrt{v_1^2 + v_2^2}$ and $S = \sqrt{G^2 + V^2}$. Then

$$U = \frac{1}{V \cdot G} \begin{bmatrix} g_1 v_2 & g_2 v_1 & g_2 v_2 & g_1 v_1 \\ g_2 v_2 & -g_1 v_1 & -g_1 v_2 & g_2 v_1 \\ -g_1 v_1 & g_2 v_2 & -g_2 v_1 & g_1 v_2 \\ -g_2 v_1 & -g_1 v_2 & g_1 v_1 & g_2 v_2 \end{bmatrix}$$

$$V = \begin{bmatrix} 0 & -\frac{v_1}{V} & -\frac{G v_2}{V S} & \frac{v_2}{S} \\ 0 & -\frac{v_2}{V} & \frac{G v_1}{V S} & -\frac{v_1}{S} \\ \frac{g_1}{G} & 0 & \frac{V g_2}{G S} & \frac{g_2}{S} \\ \frac{g_2}{G} & 0 & -\frac{V g_1}{G S} & -\frac{g_1}{S} \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} V & & & \\ & G & & \\ & & S & \\ & & & 0 \end{bmatrix}$$

The pseudoinverse is then

$$\Omega^+ = V \cdot \Sigma^+ \cdot U^T$$

where

$$\Sigma^+ = \begin{bmatrix} V^{-1} & & & \\ & G^{-1} & & \\ & & S^{-1} & \\ & & & 0 \end{bmatrix}$$

References

- [Horn and Schunck, 1981] B.K.P. Horn and B.G. Schunck, "Determining Optical Flow", *Artificial Intelligence* 17, 185-204, 1981.
- [Liu et al., 1987] Y. Liu, J. Weng, N. Ahuja, and T.S. Huang, "Estimation of Rigid Body Motion Using Straight Line Correspondences", *Proc. CVPR* (1987).
- [Longuet-Higgins, 1981] H. C. Longuet-Higgins, "A Computer Algorithm for Reconstructing a Scene from Two Projections", *Nature* 293, 133-135, 1981.
- [Longuet-Higgins and Prazdny, 1980] H. C. Longuet-Higgins and K. Prazdny, "The Interpretation of a Moving Retinal Image", *Proc. Royal Soc. London B* 208, 385-397, 1980.
- [Spetsakis and Aloimonos, 1987a] M. E. Spetsakis and J. Aloimonos, "Closed Form Solution to the Structure from Motion Problem from Line Correspondences", Technical Report CAR-TR 274, Computer Vision Laboratory, University of Maryland, College Park, 1987.
- [Spetsakis and Aloimonos, 1987b] M. E. Spetsakis and J. Aloimonos, "Closed Form Solution to the Structure from Motion Problem from Line Correspondences", *Proc. AAAI*, 1987.
- [Spetsakis and Aloimonos, 1988a] M. E. Spetsakis and J. Aloimonos, "A Multi-Frame Approach to Visual Motion Perception", Technical Report CAR-TR 407, Computer Vision Laboratory, University of Maryland, College Park, 1988.
- [Spetsakis and Aloimonos, 1988b] M. E. Spetsakis and J. Aloimonos, "Optimal Estimation of Structure from Motion from Point Correspondences in Two Frames", *Proc. ICCV*, 1988.
- [Stewart, 1973] G. W. Stewart, *Introduction to Matrix Computations*, Academic Press, 1973.
- [Strang, 1980] G. Strang, *Linear Algebra and its Applications*, Harcourt Brace Jovanovich, 1980.
- [Subbarao and Waxman, 1985] M. Subbarao and A. M. Waxman, "On the Uniqueness of Image Flow Solutions for Planar Surfaces in Motion", Technical Report CAR-TR-114, Computer Vision Laboratory, University of Maryland, College Park, 1985.
- [Tsai and Huang, 1984] R. Y. Tsai and T. S. Huang, "Uniqueness and Estimation of Three Dimensional Motion Parameters of Rigid Objects with Curved Surfaces", *IEEE Trans. PAMI* 6, 13-27, 1984.
- [Waxman and Wohn, 1987] A. Waxman and K. Wohn, "Image Flow Theory", in *Advances in Computer Vision*, ed. C. M. Brown, Erlbaum, 1987.
- [Weng et al., 1987] J. Weng, T. S. Huang, and N. Ahuja, "A Two Step Approach to Optimal Motion and Structure Estimation", *Proc. IEEE Computer Society Workshop on Computer Vision*, 1987.

FIXATION: A Direct Method for Recovery of Motion and Shape in the General Case

M. Ali Taalebi-Nezhaad
MIT Artificial Intelligence Laboratory*
545 Technology Square
Cambridge, Massachusetts 02139

Abstract

In motion vision, the problem is to find, from a sequence of time varying images, the relative rotational and translational velocities between a viewer and an environment as well as the shape of objects in the environment. This paper introduces a *direct method* called *fixation* for solving the general motion vision problem. This fixation method results in a constraint equation between translational and rotational velocities that in combination with the *Brightness-Change Constraint Equation* solves the general motion vision problem, arbitrary motion with respect to an arbitrary rigid environment.

Avoiding *correspondence* and *optical flow* has been the motivation behind the *direct methods* because both solving the *correspondence* problem, and computing the *optical flow* reliably, have proven to be rather difficult and computationally expensive.

Recently direct motion vision methods, which use the image brightness information such as temporal and spatial brightness gradients directly, have used the *Brightness-Change Constraint Equation* for solving the motion vision problem in special cases such as *Known Depth*, *Pure Translation* or *Known Rotation*, *pure Rotation*, *Planar World* and *Quadratic Patches*. In contrast to those solutions, our fixation method does not put such severe restrictions on the motion or the environment.

1 Introduction

In motion vision, the goal is to recover, from time varying images, the relative motion between a viewer and an environment as well as the structure of objects in the environment. A survey of previous literatures on machine vision is given by [Barron, 1984]. Some of the current

issues in image flow theory and motion vision are discussed by [Waxman and Wohn, 1988] and [Aloimonos and Shulman, 1989]. Much of the earlier work on recovering motion has been based on either establishing correspondences between the images of prominent features (points, lines, contours, and so on) in an image sequence (for example, [Prazdny, 1979], [Ullman, 1979; Ullman, 1980], [Longuet-Higgins, 1981], and [Aloimonos and Basu, 1986]) or establishing the velocity of points over the whole image, commonly referred to as the optical flow (for example, [Ballard and Kimball, 1983], [Bruss and Horn, 1983], and [Adiv, 1985]).

In general, identifying features here means determining gray-level corners. For images of smooth objects, it is difficult to find good features or corners. Further, the *correspondence* problem has to be solved, that is, feature points from consecutive frames have to be matched.

The computation of the local flow field exploits a constraint equation between the local brightness changes and the two components of the *optical flow*. This only gives the components of flow in the direction of the brightness gradient. To compute the full flow field, one needs additional constraints such as the heuristic assumption that the flow field is locally smooth ([Hildreth, 1984], and [Horn and Schunck, 1981]). This leads to an estimated optical flow field that is not the same as the true motion field.

Both solving the *correspondence* problem, and computing *optical flow* reliably, have proven to be rather difficult and computationally expensive. This has motivated the investigation of *direct methods* which use the image brightness information directly to recover motion.

Recently direct motion vision methods have used the *Brightness-Change Constraint Equation* (BCCE) for solving the motion vision problem in special cases such as *Known Depth* [Horn and Schunck, 1981], *Pure Translation* or *Known Rotation* [Horn and Weldon Jr., 1988, Negahdaripour and Horn, 1987b], *Pure Rotation* [Horn and Weldon Jr., 1988], *Planar World* [Negahdaripour and Horn, 1987a] and *Quadratic Patches* [Negahdaripour, 1986]. In this work, a method called *fixation* has been introduced which in combination with the BCCE solves the direct motion vision problem of arbitrary motion with respect to an arbitrary rigid environment. That is, it recovers the shape, rotational velocity and translational velocity in the general case. In contrast to the

*Support for the research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-85-K-0124.

tracking methods presented in [Aloimonos and Tsakiris, 1988; Bandopadhyay *et al.*, 1986; Sadini *et al.*, 1986; Sadini and Tistarelli, 1990], our *fixation* method is not only different but also is general. For example, [Aloimonos and Tsakiris, 1988] propose a method for tracking a target of known shape and [Bandopadhyay *et al.*, 1986] use optical flow for tracking. Also [Sadini and Tistarelli, 1990] do tracking for the special case in which the component of rotational velocity along the optical axis is zero.

A block diagram of the ideas behind this work is shown in figure 1. We start with a brief review of the BCCE in

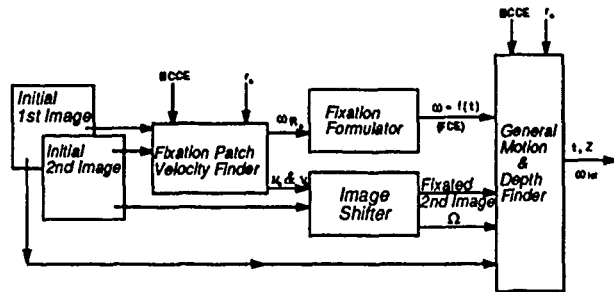


Figure 1: The *fixation* method modules.

section 2. Then in section 3, it is shown that by choosing a *fixation point* in the environment, r_0 , and knowing the component of rotational velocity along the position vector associated with that fixation point, $\omega \cdot R_0$, we can obtain a *Fixation Constraint Equation* (FCE) between the *rotational velocity* ω and the *translational velocity* t just by keeping the image of the interest point stationary in the image plane. Section 4 shows how the FCE can be combined with the BCCE and applied to *fixated images* in order to find t , ω and depth Z in the general case. Recovering $\omega \cdot R_0$, needed in the fixation constraint equation, and finding the components of *fixation velocity*, u_0 and v_0 , necessary for obtaining a fixated 2nd image, are discussed in section 5. In order to apply the FCE, a sequence of two fixated images is needed. Initial 1st image can be used directly and section 6 shows how a fixated 2nd image is obtained from the initial 2nd image.

2 The Brightness Change Constraint Equation

Using a viewer-based coordinate system which is adopted from [Longuet-Higgins and Prazdny, 1980] is very common in direct motion vision. Figure 2 depicts the coordinate system under consideration.

In this coordinate system, a world point

$$R = (X \ Y \ Z)^T \quad (1)$$

is imaged at

$$r = (x \ y \ 1)^T. \quad (2)$$

That is, the image plane has the equation $Z = 1$ or in other words the *focal length* (f) is 1. The origin is at the projection center and the Z -axis runs along the optical axis. The X - and Y - axes are parallel to the x - and y -

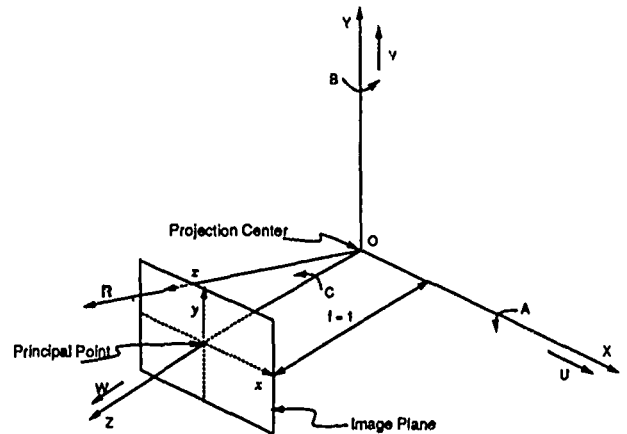


Figure 2: The viewer-centered coordinate system. The translational velocity of the camera is $t = (U \ V \ W)^T$, and its rotational velocity is $\omega = (A \ B \ C)^T$.

axes of the image plane. Image coordinates are measured relative to the *principal point*, the point $(0 \ 0 \ 1)^T$ where the optical axis pierces the image plane. The position vectors r and R are related by the perspective projection equation

$$r = (x \ y \ 1)^T = \left(\frac{X}{Z} \ \frac{Y}{Z} \ \frac{Z}{Z} \right)^T = \frac{R}{R \cdot \hat{z}} \quad (3)$$

where \hat{z} denotes the unit vector in the Z direction and $R \cdot \hat{z} = Z$.

When the observer moves with instantaneous translational velocity $t = (U \ V \ W)^T$ and instantaneous rotational velocity $\omega = (A \ B \ C)^T$ relative to a rigid environment, then the time derivative of the position vector of a point in the environment, R , relative to the viewer can be written as

$$R_t = -t - \omega \times R. \quad (4)$$

The motion of the world point R results in motion of its corresponding image point r . It can be shown [Negahdaripour and Horn, 1987a] that the motion field in the image plane is obtained by differentiating eqn. (3) with respect to time as

$$r_t = \frac{d}{dt} \left(\frac{R}{R \cdot \hat{z}} \right) = \frac{\hat{z} \times (R_t \times r)}{R \cdot \hat{z}}. \quad (5)$$

Substituting for R , r and R_t from equations (1), (2) and (4) into eqn. (5) gives, [Longuet-Higgins and Prazdny, 1980; Bruss and Horn, 1983]

$$r_t = \begin{pmatrix} x_t \\ y_t \\ z_t \end{pmatrix} = \begin{pmatrix} \frac{-U+xW}{Z} + Axy - B(x^2+1) + Cy \\ \frac{-V+yW}{Z} - Bxy + A(y^2+1) - Cx \\ 0 \end{pmatrix} \quad (6)$$

This result is just the *parallax equations of photogrammetry* that occur in the incremental adjustment of relative orientation [Hallert, 1960; Moffit and Mikhail, 1980]. It shows how, given the environment motion, the motion field can be calculated for every image point.

Image brightness changes are primarily due to the relative motion between an environment and an observer provided that the surfaces of the objects have sufficient texture and the lighting condition varies slowly enough both spatially and with time. In this case, brightness changes due to changing surface orientation and changing illumination can be neglected. Consequently, we may assume that the brightness of a small patch on a surface in the scene does not change during motion. Then expansion of the total derivative of brightness E leads to

$$\frac{dE}{dt} = E_t + x_t E_x + y_t E_y = 0^1 \quad (7)$$

which is referred to as the *Brightness Change Constraint Equation* (BCCE) [Horn and Schunck, 1981]. By substituting for x_t and y_t from eqn. (6) into eqn. (7), we obtain the brightness change constraint equation for rigid body motion [Negahdaripour and Horn, 1987a], namely

$$E_t + \mathbf{v} \cdot \boldsymbol{\omega} + \frac{\mathbf{s} \cdot \mathbf{t}}{Z} = 0 \quad (8)$$

where the auxiliary vectors \mathbf{s} and \mathbf{v} are defined as

$$\mathbf{s} = \begin{pmatrix} -E_x \\ -E_y \\ xE_x + yE_y \end{pmatrix} \quad (9)$$

and

$$\mathbf{v} = \begin{pmatrix} +E_y + y(xE_x + yE_y) \\ -E_x - x(xE_x + yE_y) \\ yE_x - xE_y \end{pmatrix}. \quad (10)$$

Considering that $\mathbf{s} \cdot \mathbf{r} = 0$, $\mathbf{v} \cdot \mathbf{r} = 0$ and $\mathbf{s} \cdot \mathbf{v} = 0$, the vectors \mathbf{r} , \mathbf{s} , and \mathbf{v} form an orthogonal triad. The vectors \mathbf{s} and \mathbf{v} represent inherent properties of the image. Also it can be shown that $\mathbf{v} = \mathbf{r} \times \mathbf{s}$. The vector \mathbf{s} indicates the directions in which translation of a given magnitude will contribute maximally to the temporal brightness change of a given picture cell. The vector \mathbf{v} plays a similar role for rotation.

The brightness change constraint equation is unchanged if we scale both Z and \mathbf{t} by the same scale factor. We conclude that we can determine only the direction of translational velocity and the relative depth of points in the scene. This well-known ambiguity is referred to as the *scale-factor ambiguity* in motion vision.

3 Fixation Formulation

Our common visual experience suggests that fixation may play an important role in the analysis of moving objects. When we want to understand the motion of an object we do not keep our eyes and head stationary in front of the moving object. Instead our head and/or eyes

¹To account for smooth variations in the image brightness due to other factors such as shading, spatial and temporal illumination changes, and variations in reflectance properties, the BCCE can be extended to

$$E_t + x_t E_x + y_t E_y = m_t E + c_t$$

where in general m_t and c_t are time and position dependent [Negahdaripour *et al.*, 1989b; Negahdaripour *et al.*, 1989a]. For simplicity, we have not used this extension here but we may use it for implementation.

follow the moving object, in order to keep the image of a point of interest stationary in the retina. There are also some formal studies that support such observations [Bahil and LaRitz, 1984; Bahil and McDonald, 1983; Bandopadhyay, 1986]. Consequently in this computer vision work, the *fixation* is defined as:

Given two subsequent images, *initial 1st* and *2nd* images, and a point in the 1st image, find a new image, *fixated 2nd image*, from the initial 2nd image such that the image of the selected point in the new image is located at its original position as in the initial 1st image.

As shown in figure 3, we refer to this selected image point as the *fixation point*, \mathbf{r}_o , and to its corresponding point on the object as the *interest point*, \mathbf{R}_o .

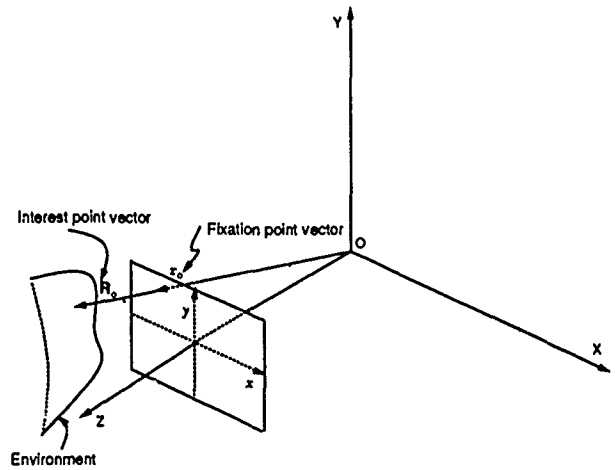


Figure 3: In fixation method, the image of the *interest point*, *fixation point*, is kept stationary in the image plane despite the relative motion between the camera and the environment.

3.1 Derivation of General Fixation Constraint Equation

For a sequence of two fixated images, at fixation point we should have

$$\mathbf{r}_{ot} = 0 \quad (11)$$

where \mathbf{r}_{ot} is the time derivative of the fixation point vector and similar to eqn. (5) can be written as

$$\mathbf{r}_{ot} = \frac{\hat{\mathbf{z}} \times (\mathbf{R}_{ot} \times \mathbf{r}_o)}{\mathbf{R}_o \cdot \hat{\mathbf{z}}}. \quad (12)$$

\mathbf{R}_{ot} is the time derivative of the interest point vector. Combination of equations (11) and (12) shows that for fixation we need to have

$$\hat{\mathbf{z}} \times (\mathbf{R}_{ot} \times \mathbf{r}_o) = 0. \quad (13)$$

In other words, we want to find out when $\mathbf{R}_{ot} \times \mathbf{r}_o$ is zero or parallel to $\hat{\mathbf{z}}$. For $\mathbf{R}_{ot} \times \mathbf{r}_o$ to be parallel to $\hat{\mathbf{z}}$, we should have \mathbf{r}_o perpendicular to $\hat{\mathbf{z}}$ which is impossible with a finite field of view (FOV), so only $\mathbf{R}_{ot} \times \mathbf{r}_o = 0$

applies. Conclusively, considering that \mathbf{R}_o and \mathbf{r}_o have the same direction, eqn. (13) is simplified as

$$\mathbf{R}_{ot} \times \mathbf{R}_o = 0 \quad (14)$$

Now substituting for $\mathbf{R}_{ot} = -\mathbf{t} - \boldsymbol{\omega} \times \mathbf{R}_o$, eqn. (4), into (14) gives

$$(\boldsymbol{\omega} \times \mathbf{R}_o) \times \mathbf{R}_o + \mathbf{t} \times \mathbf{R}_o = 0. \quad (15)$$

Expansion of eqn. (15) by using $(\mathbf{a} \times \mathbf{b}) \times \mathbf{c} = (\mathbf{c} \cdot \mathbf{a})\mathbf{b} - (\mathbf{c} \cdot \mathbf{b})\mathbf{a}$ results in

$$(\mathbf{R}_o \cdot \boldsymbol{\omega})\mathbf{R}_o - (\mathbf{R}_o \cdot \mathbf{R}_o)\boldsymbol{\omega} + \mathbf{t} \times \mathbf{R}_o = 0. \quad (16)$$

As long as the translational velocity \mathbf{t} is neither zero nor parallel to the interest point vector \mathbf{R}_o , then any vector, including $\boldsymbol{\omega}$, can be expressed in terms of the triad of vectors \mathbf{R}_o , $\mathbf{t} \times \mathbf{R}_o$ and \mathbf{t} . So we can write $\boldsymbol{\omega}$ in its general form as

$$\boldsymbol{\omega} = \alpha \mathbf{R}_o + \beta (\mathbf{t} \times \mathbf{R}_o) + \gamma \mathbf{t} \quad (17)$$

where α , β and γ are constants to be determined. Later in this section we will consider the special cases where \mathbf{t} is zero or parallel to \mathbf{R}_o by defining $\boldsymbol{\omega}$ based on another triad.

Substituting for $\boldsymbol{\omega}$ from eqn. (17) into eqn. (16) gives

$$[1 - \beta(\mathbf{R}_o \cdot \mathbf{R}_o)](\mathbf{t} \times \mathbf{R}_o) + \gamma(\mathbf{R}_o \cdot \mathbf{t})\mathbf{R}_o - \gamma(\mathbf{R}_o \cdot \mathbf{R}_o)\mathbf{t} = 0. \quad (18)$$

Now, we should find the constants β and γ such that eqn. (18) holds without putting any restrictions on \mathbf{R}_o and \mathbf{t} . We start by finding the dot product of eqn. (18) by $\mathbf{t} \times \mathbf{R}_o$ that results in

$$[1 - \beta(\mathbf{R}_o \cdot \mathbf{R}_o)]\|\mathbf{t} \times \mathbf{R}_o\|^2 = 0. \quad (19)$$

Equation (19) will hold without restricting \mathbf{R}_o and \mathbf{t} if

$$\beta = \frac{1}{\|\mathbf{R}_o\|^2}. \quad (20)$$

Another possibility for satisfying eqn. (19) is to have $\|\mathbf{t} \times \mathbf{R}_o\| = 0$ which implies that $\mathbf{t} = 0$, $\mathbf{R}_o = 0$ or \mathbf{t} is parallel to \mathbf{R}_o . But \mathbf{R}_o cannot be zero and also we assumed that here \mathbf{t} is neither zero nor parallel to \mathbf{R}_o . As a result, $\|\mathbf{t} \times \mathbf{R}_o\|$ cannot be zero. Similarly the dot product of eqn. (18) by \mathbf{t} gives

$$\gamma(\mathbf{R}_o \cdot \mathbf{t})(\mathbf{R}_o \cdot \mathbf{t}) - \gamma(\mathbf{R}_o \cdot \mathbf{R}_o)(\mathbf{t} \cdot \mathbf{t}) = 0. \quad (21)$$

Knowing that $(\mathbf{a} \times \mathbf{b}) \cdot (\mathbf{c} \times \mathbf{d}) = (\mathbf{c} \cdot \mathbf{a})(\mathbf{b} \cdot \mathbf{d}) - (\mathbf{d} \cdot \mathbf{a})(\mathbf{b} \cdot \mathbf{c})$, eqn. (21) can be simplified as

$$\gamma\|\mathbf{t} \times \mathbf{R}_o\|^2 = 0. \quad (22)$$

We discussed that $\|\mathbf{t} \times \mathbf{R}_o\|$ cannot be zero here, so eqn. (22) is satisfied only if γ is zero

$$\gamma = 0. \quad (23)$$

Substituting for β from eqn. (20) and γ from eqn. (23) into eqn. (17) gives

$$\boldsymbol{\omega} = \alpha \mathbf{R}_o + \frac{1}{\|\mathbf{R}_o\|^2}(\mathbf{t} \times \mathbf{R}_o) \quad (24)$$

where α is still unknown. This means that the component of the rotational velocity along \mathbf{R}_o cannot be

determined by the *fixation formulation*. Physically this makes sense because the rotational velocity along \mathbf{R}_o , denoted by $\omega_{\mathbf{R}_o}$, does not move the fixation point. This observation leads us to find $\omega_{\mathbf{R}_o}$ in a separate step before using the fixation formulation results. Derivation of $\omega_{\mathbf{R}_o}$ will be shown in section 5. Finally the *general fixation constraint equation* (GFCE) is written as

$$\boldsymbol{\omega} = \omega_{\mathbf{R}_o} \hat{\mathbf{R}}_o + \frac{1}{\|\mathbf{R}_o\|}(\mathbf{t} \times \hat{\mathbf{R}}_o) \quad (25)$$

where \mathbf{t} is the translational velocity and $\hat{\mathbf{R}}_o = \hat{\mathbf{r}}_o$ is the unit vector along the position vector of an arbitrary fixation point, a point in the image chosen for fixation.

3.2 Derivation of Special Fixation Constraint Equation

When the translational velocity \mathbf{t} is zero or parallel to the interest point vector \mathbf{R}_o , eqn. (16) is simplified as

$$(\mathbf{R}_o \cdot \boldsymbol{\omega})\mathbf{R}_o - (\mathbf{R}_o \cdot \mathbf{R}_o)\boldsymbol{\omega} = 0. \quad (26)$$

We define $\boldsymbol{\omega}$ based on the triad consisting of vectors \mathbf{R}_o , $\hat{\mathbf{x}}$, and $\hat{\mathbf{x}} \times \mathbf{R}_o$ as

$$\boldsymbol{\omega} = l\mathbf{R}_o + m(\hat{\mathbf{x}} \times \mathbf{R}_o) + n\hat{\mathbf{x}} \quad (27)$$

where l , m , and n are constants to be determined. Here we assume that \mathbf{R}_o is not parallel to $\hat{\mathbf{x}}$. This is a reasonable assumption because otherwise we should at least have a field of view of 180° to be able to choose an awkward interest point along the x -axis, which results in a fixation point at infinite distance from the principal point and near the border of an infinite image plane.

Substituting for $\boldsymbol{\omega}$ from eqn. (27) into eqn. (26) gives

$$[m\mathbf{R}_o \cdot (\hat{\mathbf{x}} \times \mathbf{R}_o) + n(\mathbf{R}_o \cdot \hat{\mathbf{x}})]\mathbf{R}_o - m(\mathbf{R}_o \cdot \mathbf{R}_o)(\hat{\mathbf{x}} \times \mathbf{R}_o) - n(\mathbf{R}_o \cdot \mathbf{R}_o)\hat{\mathbf{x}} = 0. \quad (28)$$

The dot product of eqn. (28) with $(\hat{\mathbf{x}} \times \mathbf{R}_o)$ results in

$$-m(\mathbf{R}_o \cdot \mathbf{R}_o)\|\hat{\mathbf{x}} \times \mathbf{R}_o\|^2 = 0. \quad (29)$$

Considering that \mathbf{R}_o cannot be either zero or parallel to $\hat{\mathbf{x}}$, eqn. (29) is satisfied only if m is zero

$$m = 0. \quad (30)$$

Substituting for m into eqn. (28) and finding its dot product by $\hat{\mathbf{x}}$ results in

$$n(\mathbf{R}_o \cdot \hat{\mathbf{x}})(\mathbf{R}_o \cdot \hat{\mathbf{x}}) - n(\mathbf{R}_o \cdot \mathbf{R}_o)(\hat{\mathbf{x}} \cdot \hat{\mathbf{x}}) = 0. \quad (31)$$

Using $(\mathbf{a} \times \mathbf{b}) \cdot (\mathbf{c} \times \mathbf{d}) = (\mathbf{c} \cdot \mathbf{a})(\mathbf{b} \cdot \mathbf{d}) - (\mathbf{d} \cdot \mathbf{a})(\mathbf{b} \cdot \mathbf{c})$, eqn. (31) can be written as

$$n\|\hat{\mathbf{x}} \times \mathbf{R}_o\|^2 = 0. \quad (32)$$

Again \mathbf{R}_o cannot be either zero or parallel to $\hat{\mathbf{x}}$. As a result, eqn. (32) will hold for arbitrary \mathbf{R}_o if $n = 0$. Substituting for n and m into eqn. (27) gives

$$\boldsymbol{\omega} = l\mathbf{R}_o \quad (33)$$

where l is still unknown. We can substitute $\omega_{\mathbf{R}_o} \hat{\mathbf{R}}_o$ for $l\mathbf{R}_o$. It will be shown later how $\omega_{\mathbf{R}_o}$ is found separately.

As a result for the special cases we obtain the *special fixation constraint equation* (SFCE) as

$$\omega = \omega_{R_0} \hat{R}_0 \quad (34)$$

which means that when the translational velocity t is zero or parallel to R_0 then the corresponding rotational velocity may only have a component along R_0 .

Our method for deriving the SFCE, eqn. (34), is not different from what we did for deriving the GFCE, eqn. (25). In fact, eqn. (34) is a special case of eqn. (25). But we could not directly derive eqn. (34) from eqn. (25) because eqn. (25) was derived based on the assumption that t is neither zero nor parallel to R_0 . As a result, for implementation it is enough to use the GFCE, eqn. (25), without knowing whether the present condition is the special case or not.

4 Solving the General Direct Motion Vision Problem

Here we assume that we are given a sequence of two fixated images. In other words we have made the fixation point stationary in the image plane. This can be done by finding the *fixation velocity*, the apparent velocity at the fixation point in the 1st image, as shown in section 5. Then the shifting method explained in section 6 can be used for generating a new image, *fixated 2nd image*, in which the fixation point is located at the same position as in the initial 1st image.

We start by studying the general case where the translational velocity t is neither zero nor parallel to the interest point vector R_0 . Then we will consider the special cases of t separately.

Substituting for ω from the general fixation constraint equation (25) into the brightness-change constraint equation (8) gives

$$E_t + \omega_{R_0} \cdot \hat{R}_0 + \frac{1}{\|R_0\|} [v \cdot (t \times \hat{R}_0)] + \frac{1}{Z} (s \cdot t) = 0. \quad (35)$$

Knowing that $a \cdot (b \times c) = (a \times b) \cdot c$ and doing some manipulations on eqn. (35) results in

$$E'_t + \left[\frac{1}{Z} s - \frac{1}{\|R_0\|} (v \times \hat{R}_0) \right] \cdot t = 0 \quad (36)$$

where E'_t is a notation for $E_t + \omega_{R_0} \cdot \hat{R}_0$ which can be computed at any pixel. In general, eqn. (36) can be solved numerically for t and Z using images of any size and with any field of view (FOV). In the following, a closed form solution is presented for the case that a small patch around the fixation point is used or the field of view is small and the whole image is used

We know that at the fixation point $v \times \hat{R}_0 = v_0 \times \hat{R}_0 = 0$. For a small field of view, the product of $v \times \hat{R}_0$ will be negligible. Even for an image with a large field of view this is still true for the image area near the fixation point. As a result, for these cases, eqn. (36) can be written as

$$E'_t + \frac{1}{Z} (s \cdot t) \approx 0 \quad (37)$$

which can be solved similar to the pure translation case [Horn and Weldon Jr., 1988, Negahdaripour and Horn,

1987b]. Questions such as "How large can the FOV or fixation patch be to guarantee the validity of this approximation?" can be answered by implementation results.

Considering that the depth range is finite, we can solve eqn. (37) by minimizing

$$\iint Z^2 dx dy = \iint \left(\frac{s \cdot t}{E'_t} \right)^2 dx dy = J \quad (38)$$

with respect to t . In other words, we are looking for the true motion t which minimizes the sum of squares of the depth estimates over the image of a scene with a finite depth range. In order to avoid the trivial solution $t = 0$, a constraint such as $\|t\| = 1$ is put on this minimization problem. This is a valid constraint on t because due to the *scale factor ambiguity* we can only find the direction of t . This constraint on t can be written as

$$t^T t = 1. \quad (39)$$

Moreover we can rewrite J as

$$J = t^T M t \quad (40)$$

where M is a fully computable 3×3 matrix

$$M = \iint \left(\frac{1}{E'_t} \right)^2 s s^T dx dy. \quad (41)$$

Minimizing J in eqn. (40) under the constraint eqn. (39) is an ordinary calculus constrained minimization problem which can be solved by minimizing

$$I(t, \lambda) = t^T M t + \lambda(1 - t^T t) \quad (42)$$

with respect to t and the Lagrange multiplier λ . Then we will have

$$\frac{\partial I}{\partial t} = 2Mt - 2\lambda t = 0 \quad (43)$$

which is simplified as

$$Mt = \lambda t. \quad (44)$$

Equation (44) is an eigenvalue problem where λ is an eigenvalue of the known matrix M and t is the corresponding eigenvector. Substituting Mt from eqn. (44) into eqn. (42) gives $I = \lambda$ which implies that under the given constraint $t^T M t$ is minimized when the smallest of three eigenvalues is used for calculating the eigenvector t .

It is concluded that the fixation method can be used for solving the motion vision problem in its general case. The translational velocity t can be calculated from eqn. (44) by using the smallest eigenvalue. Then we can use eqn. (36) for finding the environment depth

$$Z = -\frac{1}{E'_t - \frac{(v \times \hat{R}_0) \cdot t}{\|R_0\|}} (s \cdot t) \approx -\frac{1}{E'_t} (s \cdot t) \quad (45)$$

and finally eqn. (25) gives the rotational velocity ω

$$\omega = \omega_{R_0} \hat{R}_0 + \frac{1}{\|R_0\|} (t \times \hat{R}_0) \quad (46)$$

where $\|R_0\| = Z_0 \|r_0\|$ and Z_0 is obtained from eqn. (45)

The total rotational velocity of the vehicle with respect to the environment is obtained by adding ω to the *equivalent rotational velocity* Ω given in section 6. It can be seen that for the general case, the *fixation formulation* lets us find the shape and motion parameters based on an arbitrary choice of fixation point r_0 .

4.1 Special Cases: \mathbf{t} Is Zero or Parallel to \mathbf{R}_o

When the translational velocity \mathbf{t} is zero, we showed that the rotational velocity ω has only a component along \mathbf{R}_o . In this case we basically cannot obtain any estimation for the depth Z but there are methods for finding the rotational velocity ω [Horn and Weldon Jr., 1988]. For the other special case where \mathbf{t} is parallel to \mathbf{R}_o , we substitute for ω from eqn. (34) into the BCCE eqn. (8) to obtain

$$E'_t + \frac{1}{Z}(\mathbf{s} \cdot \mathbf{t}) = 0 \quad (47)$$

where E'_t is again a notation for the computable value of $E_t + \omega_{\mathbf{R}_o} \mathbf{v} \cdot \mathbf{R}_o$. Because no approximation is involved in deriving eqn. (47), an exact closed form solution exists for \mathbf{t} and Z without any restriction on the field of view or the image size. This exact solution for finding \mathbf{t} and Z is the same as the solution given in the general case, starting from eqn. (38).

5 Computing the Fixation Velocity and $\omega_{\mathbf{R}_o}$

The fixation formulation is based on the assumption that the fixation point remains stationary in a sequence of fixated images. We use the term *fixation velocity* to refer to the apparent velocity at the fixation point in the initial 1st image. We also represent x and y components of the fixation velocity by u_o and v_o respectively. The basic fixation requirement, a sequence of two fixated images in which $\mathbf{r}_{o,t} = 0$, can be satisfied by finding u_o and v_o , and then using these components for obtaining a new image, *fixated 2nd image*. The *shifting method* for obtaining the fixated 2nd image is explained in the next section.

We also saw that the component of the rotational velocity along \mathbf{R}_o , $\omega_{\mathbf{R}_o}$, cannot be obtained from the fixation formulation because this component does not move the fixation point. Here, we will introduce algorithms which can be used for finding both $\omega_{\mathbf{R}_o}$ and the components of the fixation velocity, u_o and v_o .

If we assume that depth is approximately constant on a small patch around the fixation point, the *fixation patch*, then u_o and v_o will be approximately constant on this patch. Possible sensitivity of this assumption to special cases such as slanted surfaces can be checked by implementation. Moreover the motion field velocity due to the component of the rotational velocity of the camera relative to the environment along \mathbf{R}_o is given by $-(\omega_{\mathbf{R}_o} \times \mathbf{r}) = -\omega_{\mathbf{R}_o}(\hat{\mathbf{R}}_o \times \mathbf{r}) = -\frac{\omega_{\mathbf{R}_o}}{\|\mathbf{r}_o\|}(\mathbf{r}_o \times \mathbf{r})$ because $\hat{\mathbf{R}}_o = \hat{\mathbf{r}}_o$ is the unit vector along \mathbf{r}_o . Knowing that $\mathbf{r}_o = (x_o \ y_o \ 1)^T$ and $\mathbf{r} = (x \ y \ 1)^T$, the components of the total motion field velocity along the x and y axes, due to fixation velocity and $\omega_{\mathbf{R}_o}$, are given by

$$\begin{cases} x_t = u_o - \frac{\omega_{\mathbf{R}_o}}{\|\mathbf{r}_o\|} \hat{\mathbf{x}} \cdot (\mathbf{r}_o \times \mathbf{r}) = u_o + \bar{\omega}_{\mathbf{R}_o}(y - y_o) \\ y_t = v_o - \frac{\omega_{\mathbf{R}_o}}{\|\mathbf{r}_o\|} \hat{\mathbf{y}} \cdot (\mathbf{r}_o \times \mathbf{r}) = v_o - \bar{\omega}_{\mathbf{R}_o}(x - x_o) \end{cases} \quad (48)$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ are the unit vectors along the x and y axes and $\bar{\omega}_{\mathbf{R}_o}$ is a notation for $\frac{\omega_{\mathbf{R}_o}}{\|\mathbf{r}_o\|}$. Substituting for

x_t and y_t from the above equations into the BCCE, eqn. (7), gives

$$[u_o + \bar{\omega}_{\mathbf{R}_o}(y - y_o)]E_x + [v_o - \bar{\omega}_{\mathbf{R}_o}(x - x_o)]E_y + E_t = 0. \quad (49)$$

Due to noise, eqn. (49) does not necessarily hold for any \mathbf{r} so we try to find u_o , v_o and $\bar{\omega}_{\mathbf{R}_o}$ by minimizing the sum of squares of errors over the fixation patch, denoted by p . In other words we want to minimize

$$\iint_p [(u_o + \bar{\omega}_{\mathbf{R}_o}(y - y_o)]E_x + [v_o - \bar{\omega}_{\mathbf{R}_o}(x - x_o)]E_y + E_t]^2 dx dy \quad (50)$$

with respect to u_o , v_o and $\bar{\omega}_{\mathbf{R}_o}$ which results in a system of three linear equations which can be solved for the three unknowns

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{pmatrix} u_o \\ v_o \\ \bar{\omega}_{\mathbf{R}_o} \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}. \quad (51)$$

Matrix \mathbf{A} is symmetric and its elements are given by

$$\begin{cases} a_{12} = a_{21} = \iint_p E_x E_y dx dy \\ a_{13} = a_{31} = \iint_p E_x [E_x(y - y_o) - E_y(x - x_o)] dx dy \\ a_{23} = a_{32} = \iint_p E_y [E_x(y - y_o) - E_y(x - x_o)] dx dy \\ a_{11} = \iint_p E_x^2 dx dy \\ a_{22} = \iint_p E_y^2 dx dy \\ a_{33} = \iint_p [E_x(y - y_o) - E_y(x - x_o)]^2 dx dy \end{cases} \quad (52)$$

and the elements of vector \mathbf{C} are as follows:

$$\begin{cases} c_1 = -\iint_p E_t E_x dx dy \\ c_2 = -\iint_p E_t E_y dx dy \\ c_3 = -\iint_p E_t [E_x(y - y_o) - E_y(x - x_o)] dx dy. \end{cases} \quad (53)$$

Considering that the fixation point coordinates x_o and y_o are known, then the sets of equations (52) and (53) show that the elements of matrix \mathbf{A} and vector \mathbf{C} can be calculated easily.

In the special case where the fixation point is at the principal point, $x_o = y_o = 0$, elements of \mathbf{A} are simplified

$$\begin{cases} a_{12} = a_{21} = \iint_p E_x E_y dx dy \\ a_{13} = a_{31} = \iint_p E_x (y E_x - x E_y) dx dy \\ a_{23} = a_{32} = \iint_p E_y (y E_x - x E_y) dx dy \\ a_{11} = \iint_p E_x^2 dx dy \\ a_{22} = \iint_p E_y^2 dx dy \\ a_{33} = \iint_p (y E_x - x E_y)^2 dx dy \end{cases} \quad (54)$$

and components of \mathbf{C} are given as follows:

$$\begin{cases} c_1 = -\iint_p E_t E_x dx dy \\ c_2 = -\iint_p E_t E_y dx dy \\ c_3 = -\iint_p E_t (y E_x - x E_y) dx dy. \end{cases} \quad (55)$$

After finding $\bar{\omega}_{\mathbf{R}_o}$, we can easily calculate $\omega_{\mathbf{R}_o}$ as

$$\omega_{\mathbf{R}_o} = \bar{\omega}_{\mathbf{R}_o} \sqrt{x_o^2 + y_o^2 + 1}. \quad (56)$$

When the fixation point is at the principal point, $\omega_{\mathbf{R}_o}$ is exactly the same as $\bar{\omega}_{\mathbf{R}_o}$.

6 Obtaining a Fixated 2nd Image

The fixation method assumes that a sequence of two images are available in which the fixation point is kept stationary. Referring to figure 1, we are given two initial images. The *initial 1st image* is used directly but we need to find a *fixated 2nd image*.

Physical rotation of the camera with respect to the vehicle is a hardware solution to this problem which is basically a tracking problem. Considering that in general the interest point has a relative motion with respect to the vehicle, the fixated 2nd image cannot be obtained in one step. As a result, a feedback loop is required for the camera rotation system to compensate for the errors resulted from the new position of the fixation point. This hardware approach is avoided not only because of the errors involved but also because of the concern about the real time applications.

In the following we will show how a fixated 2nd image can be obtained by applying a compensating rotation to the initial 2nd image through software. It is assumed that the fixation velocity has been already computed from eqn. (51). We introduce an *equivalent rotational velocity* $\Omega = (\Omega_x, \Omega_y, \Omega_z)$ which could result in the same fixation velocity (u_o, v_o) at the fixation point (x_o, y_o) . According to eqn. (6), the components of Ω must satisfy the following set of equations:

$$\begin{cases} u_o = x_o y_o \Omega_x - (x_o^2 + 1) \Omega_y + y_o \Omega_z \\ v_o = (y_o^2 + 1) \Omega_x - x_o y_o \Omega_y - x_o \Omega_z \end{cases} \quad (57)$$

Among infinite number of Ω that satisfy the system of equations (57), we choose the only one that does not result in any rotational velocity along the fixation point vector \mathbf{r}_o . Mathematically this means that $\Omega \cdot \mathbf{r}_o = 0$ which results in the following constraint on the components of Ω

$$x_o \Omega_x + y_o \Omega_y + \Omega_z = 0. \quad (58)$$

Given the fixation velocity (u_o, v_o) and the fixation point coordinates x_o and y_o , the equivalent rotational velocity Ω is obtained by solving the combination of three linear equations in (57) and (58). In the case that the fixation point is at the principal point, $x_o = y_o = 0$, the equivalent rotational velocity is

$$\Omega = (v_o, -u_o, 0). \quad (59)$$

Let's define the *rotational fixation velocity* as

$$\Omega_o = (\Omega_{x_o}, \Omega_{y_o}, \Omega_{z_o}) = -\Omega. \quad (60)$$

In other words Ω_o is equal to but in opposite direction of the equivalent rotational velocity Ω given by equations (57) and (58). The 2nd fixated image can be obtained by applying Ω_o to the initial 2nd image. Considering eqn. (57), the following set of equations must be satisfied in the shifting process of the initial 2nd image

$$\begin{cases} u = xy\Omega_{x_o} - (x^2 + 1)\Omega_{y_o} + y\Omega_{z_o} \\ v = (y^2 + 1)\Omega_{x_o} - xy\Omega_{y_o} - x\Omega_{z_o} \end{cases} \quad (61)$$

Here Ω_{x_o} , Ω_{y_o} and Ω_{z_o} are known, as a result the *shifting vector* (u, v) can be obtained for every pixel of the initial 2nd image. The brightness at pixel (x, y) of the

fixated 2nd image is obtained by finding the brightness at the corresponding original point $(x - Tu, y - Tv)$ in the initial 2nd image. Where T is the time interval between two initial images. In general a computed original point is not located at the center of a pixel in the initial 2nd image. As a result, its brightness cannot be read directly from the image file and should be computed by averaging, bilinear interpolation or bicubic interpolation of the brightnesses at its neighboring pixels.

7 Conclusions and Future Work

The algorithms and formulations presented in this *fixation method* show how to solve the motion vision problem directly for arbitrary motion relative to an arbitrary rigid scene. In contrast to previous work done in the area of motion vision, this solution is general and does not put any severe restriction on the motion or the shape of environment. More importantly the fixation method uses neither *optical flow* nor *feature correspondence*; instead direct image information such as temporal and spatial brightness gradients are used. There is no restriction on choosing the *fixation point*. However using the principal point as the fixation point makes the equations more concise and the calculations easier.

Implementation of this fixation method, which will be our next work, is essential for supporting the feasibility of the scheme. Referring to fig. 1, we can implement the fixation method in the following steps.

STEP 1: Finding the *fixation velocity* components (u_o, v_o) and the component of rotational velocity along $\mathbf{R} = \omega_{\mathbf{R}_o}$, by applying the system of eqn. (51) to the image information from two *initial images*.

STEP 2: Knowing the fixation velocity components, u_o and v_o , the *fixated 2nd image* is obtained by the *shifting method* explained in section 6.

STEP 3: Using the general fixation constraint equation (25), original 1st image, and the fixated 2nd image, the method presented in section 4 can be used for recovering the depth, the translational velocity and the partial rotational velocity. Note that we had to derive the special fixation constraint equation (34) separately, but for implementation it is enough to use just the general fixation constraint equation (25) because eqn. (34) is a special case of eqn. (25). As a result, the general algorithms can be used for recovering the motion and depth, without knowing in advance whether the present condition is a special case or not.

STEP 4: The *total rotational velocity* ω_{tot} is simply obtained by adding the *equivalent rotational velocity* Ω , from equations (57) and (58), to the *partial rotational velocity* ω from eqn. (46).

8 Acknowledgement

I would like to express my sincere gratitude to Prof. Berthold K. P. Horn who patiently went through several versions of this paper and provided me with many valuable ideas. Also many thanks to Prof. Ellen C. Hildreth for her crucial, constructive, and very useful comments on this work.

References

- [Adiv, 1985] G. Adiv. Determining 3-d motion and structure from optical flow generated by several moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(4):384-401, 1985.
- [Aloimonos and Basu, 1986] John Aloimonos and A. Basu. Determining the translation of a rigidly moving surface, without correspondence. In *Proceedings of IEEE Conf. on Computer Vision and Pattern Recognition*, Miami, FL, June 1986.
- [Aloimonos and Shulman, 1989] John Aloimonos and David Shulman. *Integration of Visual Modules, An extension of the Marr paradigm*. Academic Press, Boston, 1989.
- [Aloimonos and Tsakiris, 1988] John Aloimonos and D. P. Tsakiris. On the mathematics of visual tracking. Technical Report CAR-TR-390, Computer Vision Laboratory, University of Maryland, Maryland, September 1988.
- [Bahil and LaRitz, 1984] A. T. Bahil and T. LaRitz. Why can't batters keep their eyes on the ball. *American Scientist*, 72:219-253, 1984.
- [Bahil and McDonald, 1983] A. T. Bahil and J. D. McDonald. Model emulates human smooth pursuit system producing zero-latency target tracking. *Biological Cybernetics*, 48:213-222, 1983.
- [Ballard and Kimball, 1983] D. H. Ballard and O. A. Kimball. Rigid body motion from depth and optical flow. *Computer Vision, Graphics, and Image Processing*, 22(1), April 1983.
- [Bandopadhyay et al., 1986] A. Bandopadhyay, B. Chandra, and D. H. Ballard. Active navigation: Tracking an environmental point considered beneficial. In *Proc. of IEEE Workshop on Motion: Representation and Analysis*, pages 23-29, Kiawah Island, May 7-9 1986.
- [Bandopadhyay, 1986] Amit Bandopadhyay. *A Computational Study of Rigid Motion Perception*. PhD thesis, Department of Computer Science, University of Rochester, 1986.
- [Barron, 1984] J. Barron. A survey of approaches for determining optical flow, environmental layout and ego-motion. Technical Report RBCV-TR-84-5, University of Toronto, Ontario, Canada, November 1984.
- [Bruss and Horn, 1983] A. R. Bruss and B. K. P. Horn. Passive navigation. *Computer Vision, Graphics, and Image Processing*, 21(1):3-20, January 1983.
- [Hallert, 1960] B. Hallert. *Photogrammetry*. McGraw Hill, New York, NY, 1960.
- [Hildreth, 1984] Ellen C. Hildreth. *The Measurement of Visual Motion*. MIT Press, Cambridge, Mass., 1984.
- [Horn and Schunck, 1981] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185-203, 1981.
- [Horn and Weldon Jr., 1988] Berthold K. P. Horn and E. J. Weldon Jr. Direct methods for recovering motion. *International Journal of Computer Vision*, 2:51-76, 1988.
- [Longuet-Higgins and Prazdny, 1980] H. C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. In *Proceedings of the Royal Society of London B*, pages 385-397, 1980. B 208.
- [Longuet-Higgins, 1981] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133-135, 1981.
- [Moffit and Mikhail, 1980] F. Moffit and E. M. Mikhail. *Photogrammetry*. Harper & Row, New York, NY, 3rd edition, 1980.
- [Negahdaripour and Horn, 1987a] S. Negahdaripour and Berthold K.P. Horn. Direct passive navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):168-176, January 1987.
- [Negahdaripour and Horn, 1987b] S. Negahdaripour and Berthold K.P. Horn. Using depth-is-positive constraint to recover translational motion. In *IEEE Workshop on Computer Vision*, Miami, FL, 1987.
- [Negahdaripour et al., 1989a] Shahriar Negahdaripour, S. Lee, and A. Shokrollahi. Robust motion recovery: The two-plane method. In *Proc. of Second Inter. Conference on Intelligent Autonomous Systems*, Amsterdam, December 1989.
- [Negahdaripour et al., 1989b] Shahriar Negahdaripour, A. Shokrollahi, and M. Gennert. Relaxing the brightness constancy assumption in computing optical flow. In *Proc. of Inter. Conf. on Image Processing*, pages 806-810, Singapore, September 1989.
- [Negahdaripour, 1986] Shahriar Negahdaripour. *Direct Methods for Structure from Motion*. PhD thesis, Mechanical Engineering, MIT, September 1986.
- [Prazdny, 1979] K. Prazdny. Motion and structure from optical flow. In *Proc. of the Sixth Inter. Joint Conf. on Artificial Intelligence*, Tokyo, Japan, August 1979.
- [Sadini and Tistarelli, 1990] Giulio Sadini and Massimo Tistarelli. Active tracking strategy for monocular depth inference over multiple frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):13-27, January 1990.
- [Sadini et al., 1986] Giulio Sadini, V. Tagliasco, and Massimo Tistarelli. Analysis of object motion and camera motion in real scenes. In *Proc. of IEEE Inter. Conf. on Robotics and Automation*, pages 627-633, San Francisco, CA, April 1986.
- [Ullman, 1979] Shimon Ullman. *The Interpretation of Visual Motion*. MIT Press, Cambridge, MA, 1979.
- [Ullman, 1980] Shimon Ullman. The effect of similarity between line segments on the correspondence strength in apparent motion. *Perception*, 9:617-626, 1980.
- [Waxman and Wohn, 1988] Allen M. Waxman and Kwangyeon Wahn. *Image Flow Theory: A Framework for 3-D Inference from Time-Varying Imagery*, volume 1 of *Advances in Computer Vision*, chapter 3, pages 165-224. 1988.

Direct Non-Linear Methods for Recovering Structure and Motion

David J. Michael
MIT Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA 02139

Abstract

We discuss a non-linear least squares technique to recover both rigid-body camera motion—translation and rotation—and depth directly from time-varying imagery. No assumptions are made about spatial smoothness. Neither correspondences nor optical flow are computed or required. The motion and scene can be arbitrary (within the usual confines of sampling theory). Three image frames are used, but the motion between frames one and two does not have to be the same as that between frames two and three. We only assume that the same surface is visible in all three frames.

We present the algorithm and some preliminary experimental results with synthetic 8 bit images.

1 Introduction

Estimating the structure of a scene and the motion of a moving camera from the time-varying imagery is an inherently *non-linear* problem when no *special structure* is assumed for the scene or the motion. By *special structure*, we mean assumptions such as that the scene is planar (or very smooth) or that the motion is purely translational (or purely rotational) or that we are able to match features in successive image frames.

In this paper, we accept the non-linearity of the equations and demonstrate experimentally that standard techniques for solving non-linear least squares and unconstrained optimization problems can be applied in this case.

We use the direct approach of Horn and Weldon [Horn and Weldon, 1988], avoiding the problems of estimating optical flow and of finding correspondences. We look for solutions which minimize the deviation—in a least squares sense—from the *brightness change constraint equation* at each picture cell. In addition, we discuss how to incorporate the physically meaningful constraints of *small unknown applied linear forces*, *small unknown applied torques* and *depth is always positive*.

Since we are not restricting ourselves to special classes of imagery we cannot prove that the algorithm always converges. In fact, we expect the algorithm not to converge (or find the global minima) in cases where the im-

ages cover a small field of view or lack texture [Horn and Weldon, 1988]. We do, however, demonstrate that the algorithm is robust in the presence of quantization noise using synthetic (8 bit) image sequences. While we use three image frames, we *do not* assume that the motion between frames one and two is identical to that between two and three.

The properties of this structure-from-motion algorithm can be summarized as follows:

- No spatial smoothness is assumed (neighboring pixels are not assumed to lie on the same surface).
- No correspondences are computed or required.
- No optical flow is computed or required.
- The brightness change constraint equation is employed at three image frames (two time instants).
- The method requires solving a non-linear least squares problem.
- A dynamic model of the camera motion can be included.
- The method identifies places in the image where depth cannot be estimated accurately.

2 Motion Vision

The passive navigation paradigm in motion vision involves estimating at each instant of time the six motion parameters of a moving camera in an unknown static scene from a time sequence of images with N picture cells in each image. Three of the motion parameters estimate rotational velocities and three estimate translational velocities (only five of the parameters are unique because of a scale factor ambiguity). After the motion parameters have been obtained, it is straightforward to obtain a depth map of the scene from the imagery. (Although in practice, it is difficult to obtain motion estimates without an accurate depth map—this is the *motion vision paradox*.)

Researchers currently approach the problem with three different techniques:

1. *Discrete Methods* involve finding corresponding features in successive image frames and obtaining a transform that maps the features from one frame to the next [Longuet-Higgins, 1981]. This transform

contains the motion parameters. The main drawback to these methods is that finding robust matching features (solving the *feature detection* and *correspondence problems*) is very difficult, in general. Another drawback is that since these methods rely on data from such a small area in the image (at the features), they are very sensitive to noise [Weng *et al.*, 1989].

2. *Optical Flow Methods* use the 2-dimensional motion field as a starting point in estimating motion parameters. Here, the problem is that accurately estimating dense optical flow from imagery is still an open problem in computer vision and may be more difficult than the problem described here. Sparse estimation of optical flow is equivalent to solving the correspondence problem mentioned above.
3. *Direct Methods* estimate motion directly from image intensities and their first spatial and temporal derivatives in the whole image. This is the formulation that we use. So far, only the special (linear) cases of pure translation, pure rotation, known rotation, known depth, and planar and quadratic scenes have been considered. Here, we will discuss the general case.

3 Direct Methods

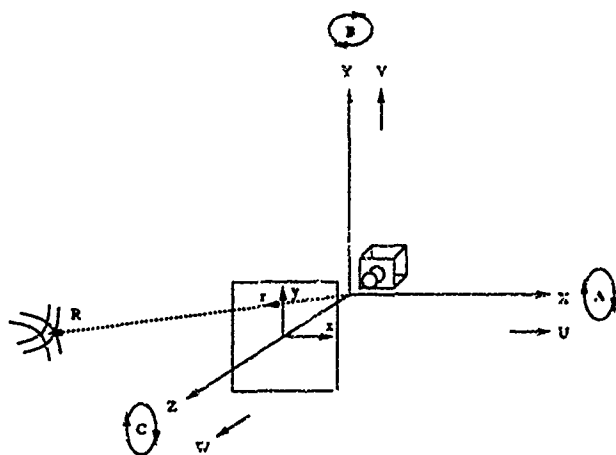


Figure 1: Perspective projection and viewer centered coordinate frame.

We assume perspective projection with the optical axis of the camera along the positive Z axis of the viewer or camera-centered coordinate frame (see figure 1) and without loss of generality assume the image $r = (x \ y \ 1)^T$ is at $Z = 1$ (corresponding to a focal length of 1) then

$$r = \frac{R}{R \cdot \hat{z}} \quad (1)$$

Thus, points in the world $R = (X \ Y \ Z)^T$ are imaged on the image plane at coordinates $(\frac{X}{Z} \ \frac{Y}{Z} \ 1)^T$. Superscript T denotes taking the transpose.

We consider instantaneous rigid motion of the camera

$$R_t = -t - \omega \times R. \quad (2)$$

where $t = (U \ V \ W)^T$ and $\omega = (A \ B \ C)^T$ are the rigid body translation and rotation parameters, respectively, corresponding to instantaneous velocities along and about the X , Y , and Z axes, respectively.

Direct methods rely on the brightness change constraint equation (BCCE) [Horn and Weldon, 1988] that relates the temporal and spatial brightness derivatives at a single image plane location and time to the motion parameters and depth of the world (corresponding to that image plane location):

$$E_t + \frac{1}{Z} s \cdot t + v \cdot \omega = 0 \quad (3)$$

where $E(x, y, t)$ is the image brightness at the point $(x \ y \ 1)$ on the image plane at time t . E_t is the temporal derivative of the image brightness. The vectors

$$s = \begin{bmatrix} -E_x \\ -E_y \\ xE_x + yE_y \end{bmatrix} \quad \text{and} \quad v = \begin{bmatrix} xyE_x + (y^2 + 1)E_y \\ -(x^2 + 1)E_x - xyE_y \\ yE_x - xE_y \end{bmatrix}$$

are simple functions of the spatial derivatives and position on the image plane. Z is the depth or Z coordinate in the camera-centered coordinate frame corresponding to $(x \ y \ 1)$ on the image plane. This constraint equation should be satisfied at every location in the image and assumes that the brightness of a particular patch in the world does not change when the camera moves. This assumption is reasonable in the case of passive navigation.

4 Places Where Depth is Inaccurate

It is important to note that in the BCCE above, the depth, Z , always appears in a product term with $s \cdot t$. This means that wherever s and t approach perpendicular or their dot product is otherwise small, the depth will have no effect on the BCCE. And, therefore, solving the BCCE at such a point will not give us reliable information about depth.

In other words, we expect *holes* in our depth map—certain places where we expect depth to be inaccurate—and we will know where these places are once we estimate t .

Also note that depth Z , and translation t , can be scaled by a constant factor without effecting the BCCE. This is the *scale factor ambiguity* in motion vision.

5 Formulation

5.1 BCCE in Time

If each pair of images has N pixels at time t_i , then N BCCE equations can be constructed (at the time instant halfway between the image frames), each of which contains a (potentially) different, independent depth, $Z_i(t_i)$, and the same six camera motion parameters, $\omega(t_i)$, the rotational velocities, and $t(t_i)$, the translational velocities. Taking into account the scale factor ambiguity, this adds up to $N + 5$ unknowns and N equations.

The BCCE can be rewritten to make the time and spatial dependencies explicit. Consider it at time t_i at picture element j .

$$\Phi_j(t_i) = E_{t,j}(t_i) + \frac{1}{Z_j(t_i)} s_j(t_i) \cdot t(t_i) + v_j(t_i) \cdot \omega(t_i) = 0 \quad (4)$$

Recall that $E_i(t_i)$, $s(t_i)$, and $v(t_i)$ are intrinsic properties of the image pair at each picture element.

At a single picture element j , we can evaluate Φ_j at time t_i and time $t_i + \Delta t$. When Δt is small, and the same surface is visible in the third frame we can relate the equations at the two time instants.

5.2 Taylor Series Expansion

The unknowns Z_j , t , and ω can be expanded in terms of Taylor series as functions of time

$$\begin{aligned} Z(t_i + \Delta t) &= Z(t_i) + \dot{Z}(t_i)(\Delta t) + \frac{1}{2}\ddot{Z}(t_i)(\Delta t)^2 + \dots \\ t(t_i + \Delta t) &= t(t_i) + \dot{t}(t_i)(\Delta t) + \frac{1}{2}\ddot{t}(t_i)(\Delta t)^2 + \dots \\ \omega(t_i + \Delta t) &= \omega(t_i) + \dot{\omega}(t_i)(\Delta t) + \frac{1}{2}\ddot{\omega}(t_i)(\Delta t)^2 + \dots \end{aligned} \quad (5)$$

We use the rigid motion assumption from equation (2) to estimate $\dot{Z}(t_i)$.

$$\dot{Z} = Z_t = \mathbf{R}_t \cdot \dot{\mathbf{z}} = (-t - \omega \times \mathbf{R}) \cdot \dot{\mathbf{z}} = -W + AY - BX \quad (6)$$

This is not the whole story. The images used at the second time instant must be dynamically *resampled* or *unwarped* to align with the images used at the first time step. This is necessary for consistency.

The physics of the particular moving camera platform can be used to find equations for \dot{t} and $\dot{\omega}$ if it is well-characterized. The equations may be analytic or merely a big look-up table. The following is an example of analytic equations. It would be appropriate for a camera mounted on a satellite.

If the camera plus satellite has mass m , Newton's second law relates the object's change-in-velocity (acceleration) to the forces applied to it and its mass [Symon, 1971],

$$m\dot{\mathbf{t}} = \mathbf{F} \quad (7)$$

This can be rewritten as

$$\dot{\mathbf{t}} = \frac{\mathbf{F}}{m} \quad (8)$$

where \mathbf{F} is the applied (linear) force. The equation of motion for the rotation of a rigid body relates the change in angular momentum of the rigid body with the applied torque

$$\frac{d\mathbf{L}}{dt} = \mathbf{N} \quad (9)$$

where $\mathbf{L} = \mathbf{I}\omega$ is the angular momentum, \mathbf{I} is the inertia tensor, and \mathbf{N} is the applied torque. Thus,

$$\dot{\omega} = -\mathbf{I}^{-1}\omega \times \mathbf{I}\omega + \mathbf{I}^{-1}\mathbf{N} \quad (10)$$

The inertia tensor may be diagonalized by choosing as coordinate axes the principal directions of inertia, and the components can be written out in slightly simpler form

$$\dot{\omega} = \begin{bmatrix} N_1/I_1 - \omega_3\omega_2(I_3 - I_2)/I_1 \\ N_2/I_2 - \omega_1\omega_3(I_1 - I_3)/I_2 \\ N_3/I_3 - \omega_2\omega_1(I_2 - I_1)/I_3 \end{bmatrix} \quad (11)$$

Note that even in the case where $\mathbf{N} = 0$, that is, where there is no applied torque, $\dot{\omega}$ is not zero. In other words, rotation is not constant in time unless the camera is constrained (special, non-zero choice of \mathbf{N}) or symmetric

($I_1 = I_2$), or is not rotating at all. (It would appear from the equations (11) that an asymmetric camera could rotate with constant velocity about a single axis. Alas, even that configuration is mechanically unstable.)

Note that in general, \mathbf{F} and \mathbf{N} , the applied force and torque are unknown parameters that need to be estimated. Some (or all) of their components may be obtained at least approximately from other sensors or from the desired control input. These known forces and torques can be separated out from the unknown forces and torques

$$\mathbf{F} = \mathbf{F}_{\text{known}} + \mathbf{F}_{\text{unknown}} \quad (12)$$

$$\mathbf{N} = \mathbf{N}_{\text{known}} + \mathbf{N}_{\text{unknown}} \quad (13)$$

where only $\mathbf{F}_{\text{unknown}}$ and $\mathbf{N}_{\text{unknown}}$ are unknown parameters. One additional constraint that can be imposed is that these unknown forces and torques are small. (If we could stochastically model these unknown forces and torques, we could recast the problem into the framework of optimal estimation.)

The known forces and torques propagate through to \dot{t} and $\dot{\omega}$, so

$$\dot{t} = \dot{t}_{\text{known}} + \dot{t}_{\text{unknown}} \quad (14)$$

$$\dot{\omega} = \dot{\omega}_{\text{known}} + \dot{\omega}_{\text{unknown}} \quad (15)$$

If we ignore second order and higher terms from the Taylor series expansion, we can rewrite the brightness change constraint equation (3) at time $t_i + \Delta t$ as

$$\begin{aligned} \Psi_j(t_i + \Delta t) &= E_{t,j}(t_i + \Delta t) \\ &+ \frac{1}{Z_j(t_i) + \dot{Z}_j(t_i)\Delta t} s_j(t_i + \Delta t) \cdot (t(t_i) + \dot{t}(t_i)\Delta t) \\ &+ v_j(t_i + \Delta t) \cdot (\omega(t_i) + \dot{\omega}(t_i)\Delta t) = 0 \end{aligned} \quad (16)$$

where $\Delta t = t_{i+1} - t_i$.

5.3 Additional Constraints and Choice of Variables

Combining the brightness constraint equations at two times, equations (4) and (16), results in $2N$ equations and $N+11$ unknowns. The system may be constrained in the likely case that N —the number of picture elements—is greater than 11.

Additional constraints can also be imposed such as requiring depth to be positive and requiring that the unknown forces and torques be small. We will show how to impose these in the next section.

We are interested in estimating the N depth values Z_j at each pixel and the eleven degrees of freedom describing the translational and rotational velocities at two time instants. Our choice of variables for these eleven degrees of freedom, t , ω , \dot{t} , and $\dot{\omega}$ is merely convenient. Instead, we could choose a set of 11 unknowns that explicitly represents the unknown forces and torques or one that completely separates the motions at the two time instants.

6 Non-Linear Least Squares

Even if the additional constraints are ignored, it is generally impossible to satisfy all $2N$ equations exactly, due to noise in the image data and the impreciseness of the

BCCE. Instead we will seek to minimize the sum, S , of the squares of the equations at time t_i and time $t_i + \Delta t$

$$S = \sum_{j=1}^N (\Psi_j(t_i))^2 + \sum_{j=1}^N (\Psi_j(t_i + \Delta t))^2 + \sum_{j=1}^{N+6} \alpha_j p_j^2 \quad (17)$$

where p_j are penalty terms. To encourage depth to be positive, penalty terms for depth are

$$p_j = \min(0, Z_j)$$

To penalize unknown forces or torques, components of $\omega_{unknown}$ and $t_{unknown}$ can be assigned to p_j

$$p_{N+j} = \omega_{j \text{ unknown}}$$

$$p_{N+3+j} = t_{j \text{ unknown}}$$

The choices of α_j should depend on our belief in the constraints and on noise in the images.

Finding unknowns that minimize S is an unconstrained minimization problem. We implemented an iterative procedure that combines an initial *steepest-descent* with a *Quasi-Newton's method* near the solution (Levenberg-Marquardt algorithm).

7 Experiments

7.1 Overview

Some of the results presented here do not include any penalty terms ($\alpha_j = 0$). In these cases the depths were *not* constrained to be positive and unknown forces and torques were *not* required to be small. When penalty terms were included, the coefficients for unknown forces and torques were chosen as unity and $1/N$ for positive depth, respectively.

The image sequences all consist of small 8 bit images. Temporal and spatial derivatives were all estimated directly from the images using a first difference approximation [Horn, 1986].

The initial conditions were chosen carefully not to bias the solution. They always consisted of a uniform distribution of depth and no motion. The scale factor was allowed to be free and normalized out at the end. See figure 2 for an example of initial depths.

The images are small because we did not worry about the speed or efficiency of the implementation. (We are currently working on improving the implementation. It appears from the results so far that the algorithm should also be practical on larger images.)

7.2 Frontal Plane

The images are of a frontal plane covered with a sinusoidal grating in both directions. (See figure 3). This is similar to a head-on picture of a wall covered with a blurry checkerboard. The algorithm converged to the correct solution in approximately 50 iterations. Table 1 shows the resulting motion parameters. Figure 4 are the depth maps of the surface at the same scale as the initial conditions depth map. Figure 5 shows image locations where we expect depth to be less accurate. Places where depth is predicted to be inaccurate have not been removed from the depth maps.

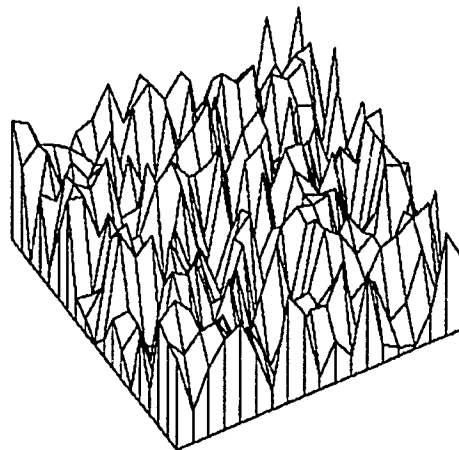


Figure 2. Initial depth map used for these experiments. Depth is chosen from a uniform distribution. (All depth maps in this paper are at the same scale.)

Without the penalty terms, excluding places where depth is estimated to be inaccurate, 63 percent of the depth estimates were within 10 percent of the correct depth. 90 percent of the depth estimates were within 17 percent of the correct depth. With the penalty terms, excluding places where depth is estimated to be inaccurate, 95 percent of the depth estimates were within 10 percent of the correct depth. 90 percent of the depth estimates were within 8.5 percent of the correct depth.

7.3 Tilted Plane

The images are of a tilted plane covered with a sinusoidal grating in both directions—much like the previous sequence, except the camera is no longer head-on to the plane. (See figure 6). The spatial frequencies of the sinusoidal grating were 33 percent higher than in the frontal

Par.	Value	Estimate
t	(3, 3, 1)	(2.9, 3.0, 1.0)
ω	(0, 0, 0)	$(1.5 \times 10^{-3}, 6.3 \times 10^{-4}, -4.1 \times 10^{-3})$
\dot{t}	(0, 0, 0)	(0.31, 0.34, 0.049)
$\dot{\omega}$	(0, 0, 0)	$(8.5 \times 10^{-4}, 3.8 \times 10^{-5}, -1.8 \times 10^{-3})$
t	(3, 3, 1)	(2.9, 3.0, 1.0)
ω	(0, 0, 0)	$(5.7 \times 10^{-4}, 1.3 \times 10^{-4}, -1.1 \times 10^{-3})$
\dot{t}	(0, 0, 0)	(0.33, 0.34, 0.053)
$\dot{\omega}$	(0, 0, 0)	$(4.9 \times 10^{-4}, -1.9 \times 10^{-4}, -5.1 \times 10^{-4})$

Table 1: Estimates of motion parameters from three 8 bit images of a frontal plane without penalty terms (top) and with penalty terms (bottom).

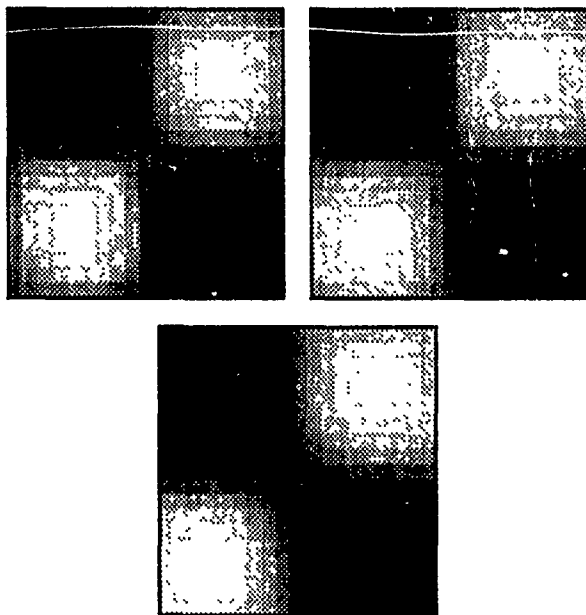


Figure 3: Three 8-bit images of frontal plane that were used as input data.

plane case causing larger spatial gradients and a smaller number of *holes* in the depth map. The algorithm converged to the correct solution in approximately 50 iterations. Table 2 shows the resulting motion parameters. Figure 7 are depth maps of the surface at the same scale as the initial conditions depth map. Figure 8 shows image locations where we expect depth to be less accurate. Places where depth is predicted to be inaccurate have not been removed from the depth maps.

The tilted plane is an interesting case because it is ambiguous [Negahdaripour and Horn, 1987]. Depending on the initial conditions, the algorithm was able to find both the solution and its dual.

Without the penalty terms, excluding places where depth is estimated to be inaccurate, 87 percent of the

depth estimates were within 10 percent of the correct depth. 90 percent of the depth estimates were within 12 percent of the correct depth. With the penalty terms, excluding places where depth is estimated to be inaccurate, 90 percent of the depth estimates were within 10 percent of the correct depth.

8 Conclusion

We demonstrated the feasibility of estimating general structure and motion from time-varying image sequences using direct methods, without imposing smoothness on the surface or linearity on the motion.

We showed that incorporating physical models of the moving camera platform can improve our structure and motion estimates.

9 Future Work

More experiments need to be done using real images. Further improvements to the implementation are necessary. The scheme can be extended using stochastic models of the camera motion.

Filling in the holes in the depth map is an important issue that we do not address here.

References

- [Horn and Weldon, 1988] B. K. P. Horn and E. J. Weldon Jr. Direct methods for recovering motion. *International Journal of Computer Vision*, 2, 1988.
- [Horn, 1986] Berthold K. P. Horn. *Robot Vision*. MIT Press, 1986.
- [Longuet-Higgins, 1981] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133-135, 1981.
- [Negahdaripour and Horn, 1987] Shariar Negahdaripour and Berthold Horn. Direct passive navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(1), 1 1987.
- [Symon, 1971] Keith R. Symon. *Mechanics*. Addison-Wesley, third edition, 1971.
- [Weng et al., 1989] Juyang Weng, Thomas Huang, and Narendra Ahuja. Motion and structure from two perspective views: Algorithms, error analysis, and error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-11(5), 5 1989.

Par.	Value	Estimate from Three Images
t	(3, 3, 1)	(3.0, 3.1, 1.0)
ω	(0, 0, 0)	$(-4.2 \times 10^{-4}, 1.0 \times 10^{-4}, 1.5 \times 10^{-3})$
\dot{t}	(0, 0, 0)	(0.34, 0.31, 0.085)
$\dot{\omega}$	(0, 0, 0)	$(-3.0 \times 10^{-4}, -1.6 \times 10^{-4}, 7.8 \times 10^{-4})$
t	(3, 3, 1)	(3.0, 3.1, 1.0)
ω	(0, 0, 0)	$(-4.9 \times 10^{-4}, -3.5 \times 10^{-5}, 1.7 \times 10^{-3})$
\dot{t}	(0, 0, 0)	(0.34, 0.32, 0.084)
$\dot{\omega}$	(0, 0, 0)	$(-3.3 \times 10^{-4}, -2.2 \times 10^{-4}, 8.4 \times 10^{-4})$

Table 2: Estimates of motion parameters from three 8 bit images of a tilted plane without penalty terms (top) and with penalty terms (bottom).

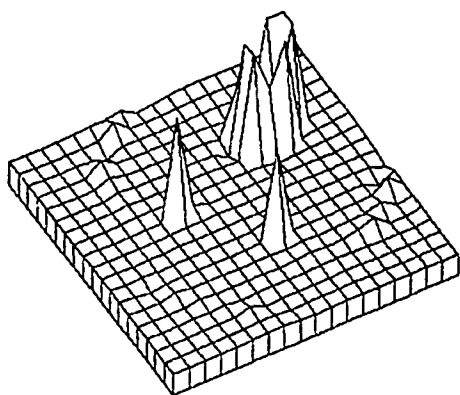
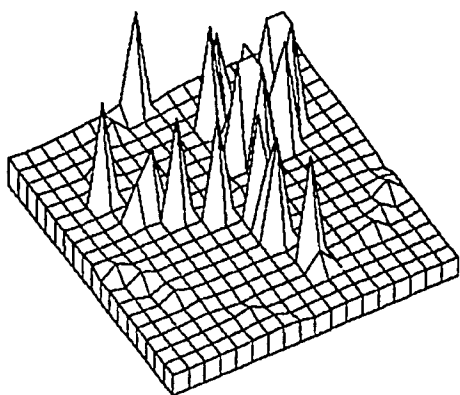


Figure 4: Estimated depth maps for the case of the frontal plane without penalty terms (top) and with penalty terms (bottom). (All depth maps in this paper are at the same scale.)

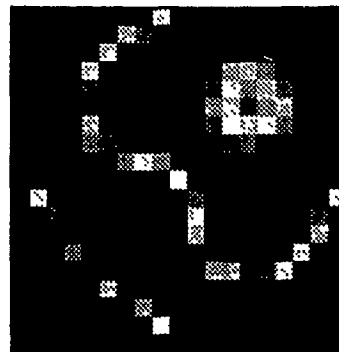


Figure 5: Places where we expect depth estimates to be inaccurate for the case of the frontal plane without penalty terms. Dark picture elements have estimated values of $|s \cdot t|$ larger than 50. The remaining picture elements are scaled to be brighter when the estimated $|s \cdot t|$ is nearer to zero.

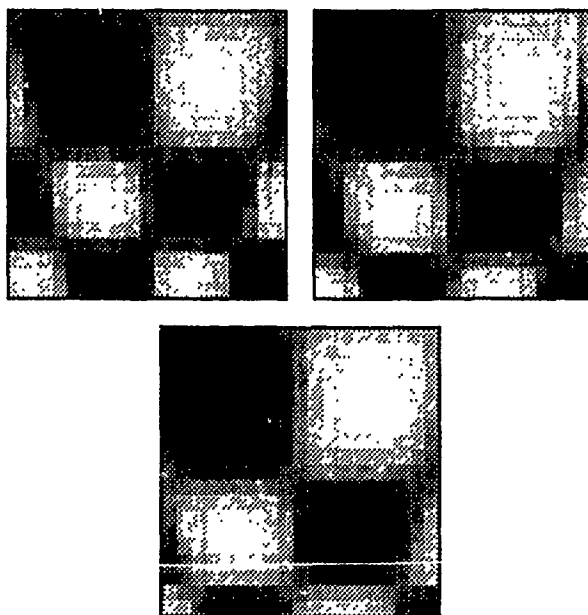


Figure 6: Three 8-bit images of tilted plane that were used as input data.

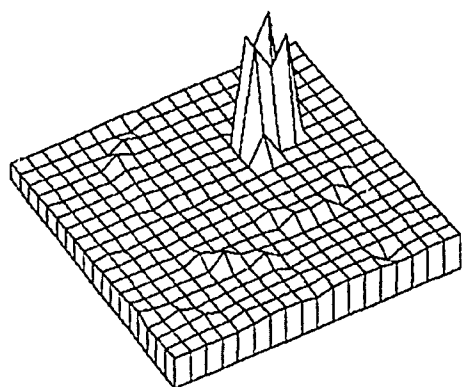
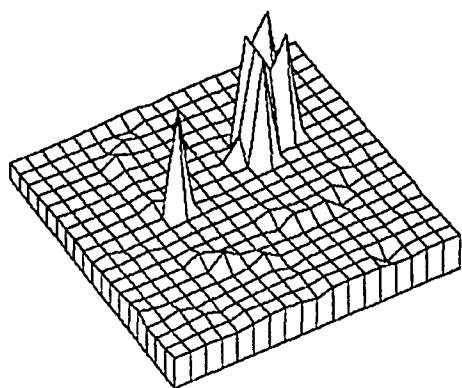


Figure 7: Estimated depth maps for the case of the tilted plane without (top) and with (bottom) penalty terms. (All depth maps in this paper are at the same scale.)

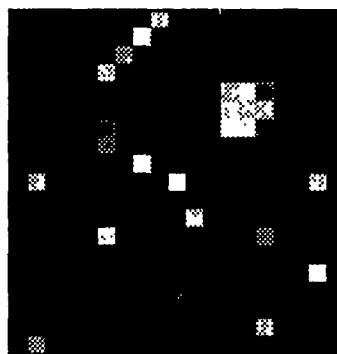


Figure 8: Places where we expect depth estimates to be inaccurate for the case of the tilted plane without penalty terms. Dark picture elements have estimated values of $|s \cdot t|$ larger than 50. The remaining picture elements are scaled to be brighter when the estimated $|s \cdot t|$ is nearer to zero.

Robustness of Correspondence-Based Structure from Motion¹

R. Dutta M. A. Snyder
Computer and Information Science Department
University of Massachusetts at Amherst
Amherst, Massachusetts 01003

Abstract

This paper examines the robustness of correspondence-based approaches to structure from motion. Unlike earlier studies it is proven in an *algorithm-independent* way, that small absolute errors in image displacements cause absolute errors in rotational motion parameters significant enough to lead to large relative errors in the determination of environmental depth. Even if the motion parameters are known exactly, through sophisticated navigation systems, small errors in image displacements still lead to large errors in depth for environmental points whose distance from the camera is greater than a few multiples of the total translation in depth of the camera.

1 Introduction

A large number of mathematically elegant approaches for determining structure from motion have relied on some form of point or feature correspondences between two or more perspective views [1-8]. These correspondence-based approaches take advantage of the image displacements induced by egomotion. Most such methods match a large number of points or features in two temporally separated images and quantitatively measure the image displacements. A consistent set of motion parameters is then determined to explain these displacements. Once the motion parameters have been determined, the depth of environmental points can be found by using their individual image displacements. In a few algorithms (e.g. [5]) optimal values of environmental depths and motion parameters are found concurrently.

In spite of the extensive research that has been conducted on this problem, robust depth determination under a wide variety of scene structures (especially in an outdoor environment) has remained elusive. Naturally, this has led to investigations into the reason for such failures. Algorithm-specific errors have been explored by Tsai and Huang [9], Barron [10], and Fang and Huang [7]. General methodologies for computing the precision of 3-D Parameters in the case of translational motion have been explored by Snyder [11]. Large shifts in the Focus of Expansion have been shown to occur in the case of approximate translational motion when small rotations have been ignored [12,13]. Weng, Huang, and

Ahuja [8] have given some qualitative conditions for better structure recovery, based primarily on simulations. Adiv [14] has characterized and demonstrated situations where inherent ambiguities exist in the interpretation of noisy flow fields. In addition to the above there are a large number of other serious studies of this problem (see the review in [15]). However, what is lacking is a comprehensive algorithm-independent study of the issues associated with the correspondence-based structure from motion problem. The present paper is an attempt to remedy this situation.

The key problem is to show, in an algorithm-independent way, the effect of all relevant quantities on the computation of environmental depth. We start by using the small rotation image displacement equations and show mathematically that small absolute errors in the determination of rotational motion parameters cause large relative errors in depth computation unless the image displacements are really very large. This is proved in an algorithm-independent way and has the immediate consequence that motion algorithms must precisely estimate rotations. The question might be raised as to whether motion algorithms can indeed estimate rotations precisely. We prove that it is not possible to do so by showing that small absolute errors in image displacements give rise to significant rotational errors. The proofs also show that the occurrence of the large errors is ubiquitous. We therefore conclude from this *theoretical* analysis that it is in general difficult to secure robust depth of environmental points. We emphasize that in contrast to our algorithm-independent theoretical approach the conclusions of the earlier studies have either been based on qualitative frameworks or have been algorithm-dependent.

In addition to the above, we explore the direct relationship between small errors in the image displacement and computed relative depth. This is necessary because of suggestions that sophisticated navigation systems can give motion parameters exactly. It is shown that even if the motion parameters are known exactly, small errors in computing the magnitude of the correct image displacement generally cause large errors in depth.

We begin our analysis by stating the small rotation image displacement equations in the next section.

2 Depth from Image Displacement

In this section we give the equations for computing depth from motion. We assume a right handed coordinate system fixed with respect to the camera as shown in Figure 1. Let us also assume that the right hand rule is used for rotations. We consider the case where the camera is undergoing motion. As can be seen from Figure 1 the

¹This research has been supported by the Defense Advanced Research Projects Agency under RADC contract F30602-87-C-0140 and Army ETL contract DACA76-89-C-0017.

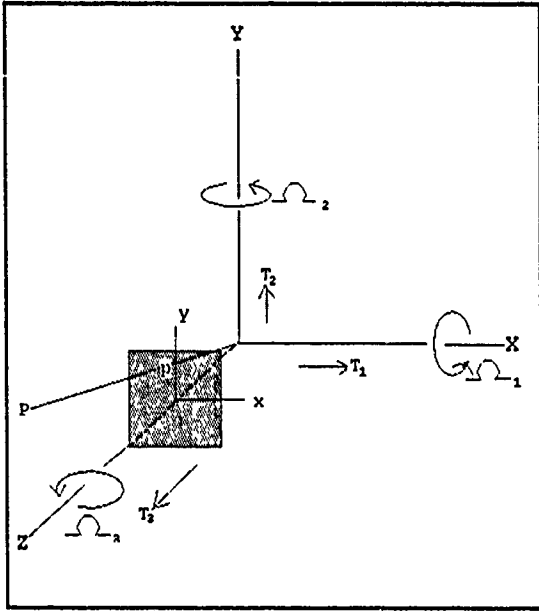


Figure 1: Coordinate System

environmental point P , with world coordinate (X, Y, Z) , is projected onto point p , in the image plane with image coordinates (x, y) . Let f be the focal length of the camera, and denote by $\vec{T} = (T_1, T_2, T_3)$, $\vec{\Omega} = (\Omega_1, \Omega_2, \Omega_3)$ the translational and rotational rigid motion of the camera (This implies that $P' = RP + T$ where R is the rotation matrix and P' is the new position of P after undergoing rigid motion).

We shall be working with the small rotation motion equations. For small rotations $\sin(\theta) \approx \theta$ and $\cos(\theta) \approx 1$.

For simplicity we define the following abbreviations:

$$I = 1 + \frac{\Omega_2 x}{f} - \frac{\Omega_1 y}{f} \quad (1)$$

$$J = \left(\frac{\Omega_1 xy - \Omega_2 x^2}{f} \right) - \Omega_2 f + \Omega_3 y \quad (2)$$

$$K = \left(\frac{\Omega_1 y^2 - \Omega_2 xy}{f} \right) + \Omega_1 f - \Omega_3 x \quad (3)$$

$$\alpha = -fT_1 + xT_3 \quad (4)$$

$$\beta = -fT_2 + yT_3. \quad (5)$$

It should be noted from the abbreviations that I, J , and K are functions of rotational parameters whereas α and β are functions of translational parameters. With the above abbreviations the image displacement \vec{l} , induced by the motion of the camera, is given by

$$\vec{l} = u \hat{x} + v \hat{y} \equiv (u, v) \quad (6)$$

where \hat{x} and \hat{y} are unit vectors along the x -axis and y -axis respectively, and ³

²With this approximation, for small angles the errors in the sine and cosine of θ are negligible. For example even if $|\theta| = 0.100$ radians (5.7°), the relative error in $\sin(\theta)$ is 0.167% and in $\cos(\theta)$ is 0.500%. It should be noted that even though we are considering small angles in the derivations, the absolute errors in rotation we shall be considering will sometimes have a least value that is roughly 600 times smaller. Besides, it will be shown later that the small angle assumption is not a restrictive one in practical situations.

³Various equivalent versions can be found in the literature

$$u = \frac{J + (\alpha/Z)}{I - (T_3/Z)} \quad (7)$$

$$v = \frac{K + (\beta/Z)}{I - (T_3/Z)}. \quad (8)$$

In addition, we choose units in which $|\vec{T}| = 1$, so that

$$T_1^2 + T_2^2 + T_3^2 = 1. \quad (9)$$

The depth Z of an environmental point P can be determined from either equation (7) or equation (8). We denote by Z_x the depth determined from equation (7) and by Z_y the depth determined from equation (8). Hence,

$$Z_x = \frac{T_3 u + \alpha}{I u - J} \quad (10)$$

$$Z_y = \frac{T_3 v + \beta}{I v - K}. \quad (11)$$

It should be noted that it is possible to write the depth in terms of the displacement vector \vec{l} and the motion parameters by using equations (6), (7) and (8). Since this becomes rather cumbersome to manipulate for analyzing errors, we shall often (but not always) work with Z_x and Z_y separately. When we use Z_x the conclusions are made using the x -component of the image displacement and when we use Z_y the conclusions are made using the y -component of the image displacement. The depth is expressed in units of the total translation, $|\vec{T}|$.

It is often convenient to measure image distances in pixels. The focal length measured in pixels given by

$$f = \frac{N}{2} \cot \left(\frac{FOV}{2} \right) \text{ pixels} \quad (12)$$

where the image dimension is $(N \text{ pixels} \times N \text{ pixels})$ and FOV is the field of view of the camera. For a 256×256 image with a 45° field of view the focal length is computed to be 309 pixels with this formula.

With these preliminaries in mind we proceed to analyze the effect of various quantities on environmental depth in the next few sections.

3 Effect of Small Rotational Errors on Depth

In this section we prove mathematically that small absolute errors in computing rotations cause very large relative errors in depth.

Equation (10) gives us the depth of an environmental point in terms of the motion parameters, camera focal length and image displacement at that point. Hence, to find the effect of small rotational errors on environmental depth we use equation (10) and take the partial derivative with respect to the rotation around the Y -axis, Ω_2 to get ⁴

[7,16]. Usually most derivations proceed by assuming f to be 1. For convenience of experimental verification we have not made this assumption.

⁴There would have been higher order terms in the expression on the right hand side of equation (13) if we had considered higher order terms in the expansion of sines and cosines of angles. However, it can be shown that for small rotations of the camera the higher order terms contribute negligibly when compared to the terms that we have considered.

$$\frac{\delta Z_x}{\delta \Omega_2} = -Z_x \left(\frac{xu}{f} + \frac{x^2}{f} + f \right) \left[\left(1 + \frac{\Omega_2 x}{f} - \frac{\Omega_1 y}{f} \right) u - \left(\frac{\Omega_1 xy - \Omega_2 x^2}{f} - \Omega_2 f + \Omega_3 y \right) \right]^{-1} \quad (13)$$

This can be rewritten as

$$\frac{\delta Z_x}{Z_x} = - \left(f^2 + x^2 + xu \right) \left[fu + \Omega_2 xu - \Omega_1 yu - \Omega_1 xy + \Omega_2 x^2 + \Omega_2 f^2 - \Omega_3 yf \right]^{-1} \delta \Omega_2. \quad (14)$$

From equation (14)

$$\frac{\delta Z_x}{Z_x} = -G(x, y) \delta \Omega_2, \quad (15)$$

where

$$G(x, y) = (f^2 + x^2 + xu) \left[fu + \Omega_2 xu - \Omega_1 yu - \Omega_1 xy + \Omega_2 x^2 + \Omega_2 f^2 - \Omega_3 yf \right]^{-1}. \quad (16)$$

Equation (14) gives the relative error in depth, due to a small error in the rotational parameter Ω_2 , at coordinate (x, y) in the image.

In section 3.1 we discuss the errors at the center of the image and in section 3.2 we derive an approximate expression for the error at all locations in the image. We begin by choosing for illustration the center of the image because the principal objects of interest are frequently projected there. Also, in applications using tracking it is a commonly used point. However, it is insufficient to show that the large errors occur only in a specific position and therefore we also search for a general solution.

3.1 Error at the center of the image

At the center of the image ($x = 0, y = 0$) the relative error in depth is found by substituting 0 for both x and y in equation (14):

$$\frac{\delta Z_x}{Z_x} = - \left(\frac{f}{u + \Omega_2 f} \right) (\delta \Omega_2). \quad (17)$$

Equation (17) indicates that we are not just considering very small rotational errors in *small* rotations because what is important is the sum of u and $\Omega_2 f$. If $(u + \Omega_2 f)$ is small, the large magnitude of Ω_2 by itself will not be able to help reduce the relative error in depth. In any case, for practical motion analysis, the rotations cannot be very large because they would induce such large image displacements that correspondence algorithms would be unable to handle them.

We shall consider three cases:

1. $\Omega_2 \approx 0$
2. $\Omega_2 \approx -\frac{u}{f}$
3. Ω_2 is a random variable, centered at 0 with standard deviation σ .

Of the three cases the third requires some clarification. Often the true value of the rotational parameter of the moving camera is not known. That is to say

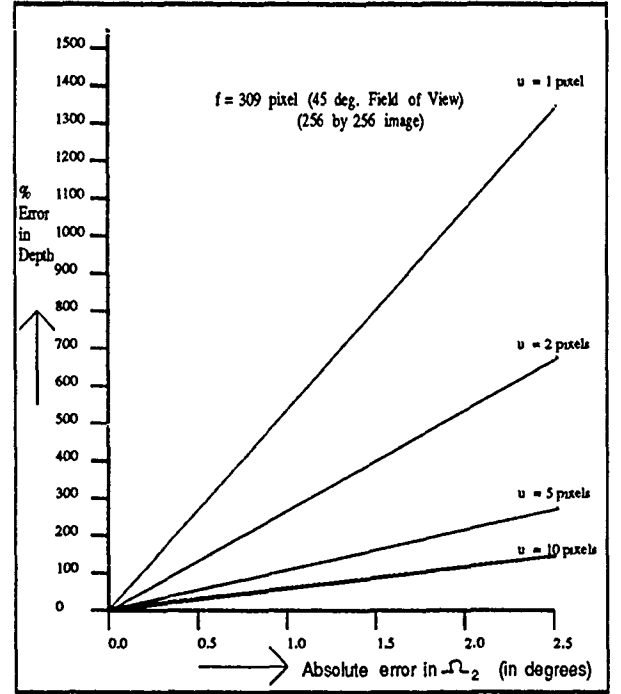


Figure 2: Relative Error in Depth

there is an uncertainty (not error) associated with it. Hence this is modelled as a random variable. When we measure Ω_2 there is some error $\delta \Omega_2$, which is the difference between the experimental and theoretical values (i.e. ground truth) of Ω_2 . It should be noted that we choose the distribution of Ω_2 so that it is similar to the distributions usually encountered for the true values of the rotational parameters of a moving camera⁵. On the other hand, $\delta \Omega_2$ is the error which a motion algorithm makes in finding the true Ω_2 .

Case 1: Ω_2 is close to 0 - When $\Omega_2 \approx 0$ equation (17) can be written as (considering only absolute values of the error)

$$\frac{\delta Z_x}{Z_x} \approx \left(\frac{f}{|u|} \right) (|\delta \Omega_2|) \quad (18)$$

Equation (18) gives the relative error in computed depth with respect to the absolute error in the computation of the rotation around the Y-axis. Note that since the image displacement, $|u|$, is necessarily much less than the focal length of the camera (recall that f is typically several hundred pixels), even a small error in Ω_2

⁵Vehicle (not camera) rotations measured by a land navigation system on the Autonomous Land Vehicle (ALV) [17] strongly suggest a bell shaped distribution for the rotation R of the ALV around the axis perpendicular to its base when it is moved approximately straight ahead. Since the ALV is 16000 pounds in weight, eight wheel powered, and hydrostatically driven, it is expected that the spread of the angles would be very small on an almost level terrain. However actual data from a 30 frame sequence collected on almost level ground has the spread of R at about 2.4° . For smaller, lighter vehicles and uneven terrain the spread is likely to be even more. In our theoretical analyses, whenever we assign specific values to the parameters for illustration, we choose the spread in such a way that it becomes apparent that we are seeking some kind lower bound on the errors.

results in a large relative error in depth.

In Figure 2 the results of using equation (18) to find the relative error in depth are shown. It is remarkable to see that we get over 100% error in depth estimation when $\delta\Omega_2$ is as small as 0.5° and the displacement vector, u is 2 pixels. We also note that even when $\delta\Omega_2$ is 0.1° and the displacement vector, u , is 2 pixels we get a 27% relative error in depth. Hence the absolute error in finding the rotational parameter Ω_2 must be very small indeed.

Case 2: Ω_2 is nearly equal to $(-u/f)$ - From equation (17) we see that when $\Omega_2 = -\frac{u}{f}$ the relative error in depth is infinite. This result should be treated with caution since we have ignored (Ω_2^2) and higher order terms in the Taylor series expansions of the sine and cosine of Ω_2 . However, as shown earlier, for small angles the errors in the sine and cosine of Ω_2 are negligible. Hence we can safely say that for a 256×256 image with a field of view of 45° the relative error in depth can become enormous for many combinations of values of the displacement vector, u and the rotational angle Ω_2 (e.g. $u = 5$ pixel and $\Omega_2 = -0.016 \text{ rad.} = -1^\circ$). When Ω_2 is nearly equal to $(-u/f)$, this means that the image displacement u is nearly equal to $(-\Omega_2 f)$. Qualitatively speaking the image displacement in this case is solely due to rotation and hence depth cannot be determined. As a corollary the larger the contribution of rotation to the image displacement the larger the error in depth.

Case 3: Ω_2 is a Random Variable - Let Ω_2 be a random variable. Then we find the probability that the relative error in depth ξ , is less than $|k|$.

With a normal distribution for Ω_2 numerical solutions are needed for computing the relative error in depth. We secure a closed form solution by considering a different probability density function, $q(\Omega_2)$ for Ω_2 which we call the Modified Inverse Square Distribution, given by

$$q(\Omega_2) = \frac{M}{\pi} \frac{1}{1 + M^2 \Omega_2^2} \quad (19)$$

From Figure 3 it can be seen that the modified inverse square distribution looks similar to the normal distribution. However it lacks some of the nice properties of the latter. The spread of the normal curve is controlled by the standard deviation, σ whereas the spread of the modified inverse square curve can be controlled by adjusting $\frac{1}{M}$ (i.e. large $\sigma \sim$ small M). For our purposes either distribution for Ω_2 would be fine.

The relative error ξ in depth is a function of the random variable Ω_2 . Using equation (17) we find that

$$\xi = \frac{1}{a\Omega_2 + b} \quad (20)$$

where

$$a = -\left(\frac{1}{\delta\Omega_2}\right), \quad b = -\frac{u}{f\delta\Omega_2} \quad (21)$$

We can solve analytically for the probability $P(\xi \leq |k|)$ that the relative error ξ in depth is less than $|k|$. This is shown in equation (22)

$$P(\xi \leq |k|) = \frac{1}{\pi} \left[\tan^{-1} \frac{(a^2 + M^2 b^2)k + M^2 b}{M|a|} + \tan^{-1} \frac{(a^2 + M^2 b^2)k - M^2 b}{M|a|} \right] \quad (22)$$

Table 1 has been constructed using equation (22) with $M = 100$. This choice of M gives a 0.94 probability that $|\Omega_2| \leq 0.1 \text{ rad.}$, a 0.87 probability that $|\Omega_2| \leq 0.05 \text{ rad.}$, and a 0.5 probability that $|\Omega_2| \leq 0.01 \text{ rad.}$. From the point of view of error analysis this kind of distribution for Ω_2 is a conservative model (i.e. we are

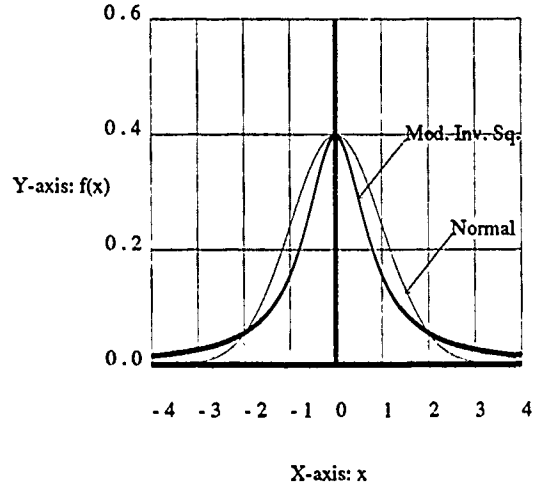


Figure 3: Normal and Modified Inverse Square density functions. Normal Distribution shown has mean = 0 and standard deviation = 1. Modified Inverse Square Density has $M = 1.25$

trying to find lower bounds) for a vehicle which is trying to move without much rotation.

From Table 1 it can be concluded that even with very small errors in rotation the relative errors in depth are very high. As an example it can be computed from the table that with an absolute error in rotation of 0.10° for displacement vectors of 5 pixel length, in 78 out of 100 cases the relative error in depth will be more than 5%, and in 45 out of 100 cases the relative error in depth will be more than 10%, and in 13 out of 100 cases the relative error will be more than 25%. For smaller image displacements like 1 or 2 pixels the results are not very good even with absolute errors in rotation of 0.01° (e.g. with an image displacement of 1 pixel, in 1 out of 5 cases the relative error is more than 5% and in 1 out of 10 cases the relative error is more than 10%).

Conclusions - We can safely conclude from the above analysis that small errors in estimating rotations can cause very large relative errors in depth computation. This conclusion is strictly true for error in rotations around the x and y axes, since analogous results can be obtained for Ω_1 by considering Z_y . Hence the absolute error in the estimation of rotational parameters, particularly Ω_1 and Ω_2 , has to be very small indeed. It should be noted that at the center of the image Ω_3 (rotation around the Z -axis) does not cause any error in depth.

3.2 General solution by approximations with Taylor Series

In section 3.1 we found the relative error in depth at the center of the image with no approximations except for ignoring second and higher order terms in the expansions of sine and cosine of small angles. It is seen from the results of section 3.1 that for small displacements the relative error in depth is high. In this section we give somewhat more general results. It should be noted that this is much harder because in order to arrive at good approximations we have to take into account the feasible range of various parameters in typical scenes.

Table 1: Probability(Relative error in depth, $\xi \leq |k|$) using Modified Inverse Square distribution (with no Taylor series approximations). The results of this table are strictly valid only for the center of the image. $M = 100$; 256×256 image; Field of View = 45° .

$\delta\Omega_2 = 0.50^\circ$		Relative error in depth, k					
displ. (abs val)		1%	5%	10%	25%	50%	100%
1 pix.		0.01	0.04	0.07	0.18	0.34	0.56
2 pix.		0.01	0.04	0.07	0.18	0.36	0.61
5 pix.		0.01	0.04	0.08	0.22	0.55	0.83
10 pix.		0.01	0.04	0.08	0.47	0.87	0.95

$\delta\Omega_2 = 0.10^\circ$		Relative error in depth, k					
displ. (abs val)		1%	5%	10%	25%	50%	100%
1 pix.		0.04	0.18	0.34	0.63	0.80	0.90
2 pix.		0.04	0.18	0.36	0.69	0.84	0.92
5 pix.		0.04	0.22	0.55	0.87	0.94	0.97
10 pix.		0.04	0.47	0.87	0.96	0.98	0.99

$\delta\Omega_2 = 0.05^\circ$		Relative error in depth, k					
displ. (abs val)		1%	5%	10%	25%	50%	100%
1 pix.		0.07	0.34	0.56	0.80	0.90	0.95
2 pix.		0.07	0.36	0.61	0.84	0.92	0.96
5 pix.		0.08	0.55	0.83	0.94	0.97	0.98
10 pix.		0.08	0.87	0.95	0.98	0.99	1.00

$\delta\Omega_2 = 0.01^\circ$		Relative error in depth, k					
displ. (abs val)		1%	5%	10%	25%	50%	100%
1 pix.		0.34	0.80	0.90	0.96	0.98	0.99
2 pix.		0.36	0.84	0.92	0.97	0.98	0.99
5 pix.		0.55	0.94	0.97	0.99	0.99	1.00
10 pix.		0.87	0.98	0.99	1.00	1.00	1.00

Let $\Psi = \frac{x}{f}$ and $\Upsilon = \frac{y}{f}$. Then the function $G(x, y)$ in equation (16) can be written as

$$G(x, y) = F(\Psi, \Upsilon) = \left(1 + \Psi^2 + \frac{u}{f}\Psi \right) \left[\frac{u}{f} + \Omega_2 \frac{u}{f}\Psi - \Omega_1 \frac{u}{f}\Upsilon - \Omega_1 \Psi \Upsilon + \Omega_2 \Psi^2 + \Omega_2 - \Omega_3 \Upsilon \right]^{-1} \quad (23)$$

We expand $F(\Psi, \Upsilon)$ in a Taylor series T_0 about $(\Psi = 0, \Upsilon = 0)$ and ignore terms involving second or higher order derivatives of F . This approximation means that we are omitting $O((\frac{x}{f})^2)$, $O((\frac{y}{f})^2)$ and higher order terms. This is a very good approximation in the central region of any image. Even in the periphery of almost all commonly collected images it is a very good approximation [e.g. with a Field of View of 45° for a 256×256 image, $f = 309$ pixels, and so $(\frac{x}{f})^2$ and $(\frac{y}{f})^2$ are about 0.17 at most, in any part of image]. Within T_0 there are some expressions which are functions of Ω_2 . We expand these functions of Ω_2 in a Taylor series about $\Omega_2 = 0$ and ignore terms with second or higher order derivatives. The larger the value of u when compared to $\Omega_2 f$ the better the approximation. Note that in section 2 we had omitted $O(\Omega_2^2)$ and higher terms while approximating $\cos(\Omega_2)$ and $\sin(\Omega_2)$ in the rotation matrices. We also omit terms which are of order greater than or equal to $O(PQ)$ where $P \in \{\Omega_1, \Omega_2, \Omega_3\}$ and $Q \in \{\Psi, \Upsilon\}$, since they are of the second order or more in smallness (i.e. they are insignificant when compared to the lower order terms). We obtain

$$\begin{aligned} F(\Psi, \Upsilon) &= \left(1 + \Psi^2 + \frac{u}{f}\Psi \right) \left[\frac{u}{f} + \Omega_2 \frac{u}{f}\Psi - \Omega_1 \frac{u}{f}\Upsilon - \Omega_1 \Psi \Upsilon + \Omega_2 \Psi^2 + \Omega_2 - \Omega_3 \Upsilon \right]^{-1} \\ \Rightarrow G(x, y) &= (f^2 + x^2 + xu) \left(fu + \Omega_2 xu - \Omega_1 yu - \Omega_1 xy + \Omega_2 x^2 + \Omega_2 f^2 - \Omega_3 yf \right)^{-1} \end{aligned}$$

$$\Rightarrow G(x, y) \approx \frac{f}{u} - \frac{f^2}{u^2}\Omega_2 + \frac{x}{f} \quad (24)$$

For a 256×256 image (x/f) is strictly less than or equal to 0.4 whereas (f/u) is about 15.4 even with a displacement u , of 20 pixels. Hence we can omit (x/f) in equation(24) with negligible error. It should be noted that u is much greater than $(\Omega_2 f)$.

Therefore, we can write

$$\begin{aligned} G(x, y) &= (f^2 + x^2 + xu) \left(fu + \Omega_2 xu - \Omega_1 yu - \Omega_1 xy + \Omega_2 x^2 + \Omega_2 f^2 - \Omega_3 yf \right)^{-1} \\ &\approx \frac{f}{u} - \frac{f^2}{u^2}\Omega_2 \end{aligned} \quad (25)$$

Using equations (14) and (25) we then can express the relative error ξ in depth as

$$\xi = \frac{\delta Z_x}{Z_x} \approx \left[-\frac{f}{u} + \left(\frac{f}{u} \right)^2 \Omega_2 \right] \delta \Omega_2 \quad (26)$$

Note that equation (26) is also an approximation to equation (17) under the conditions that the magnitude of the displacement vector u , is greater than the magnitude of $\Omega_2 f$. Hence under these conditions the results of the previous section can be extended to a large part of the image. However, for ease of analysis we shall consider equation (26) rather than equation (17).

We now find the probability that the relative error in depth ξ is less than or equal to $|k|$ where Ω_2 is a random variable with modified inverse square distribution. As an aside it should be noted that numerical computations using normal distribution gives very similar results.

Ω_2 with Modified Inverse Square Distribution
The modified inverse square distribution has already been defined in section 3.1.

Let

$$c = \left(\frac{f}{u} \right)^2 \delta \Omega_2 \quad (27)$$

$$d = -\frac{f}{u} \delta \Omega_2 \quad (28)$$

Since relative error ξ in depth is a linear function of the random variable Ω_2 [from equation (26)] it is relatively easy to arrive at the following result

$$P(\xi \leq |k|) = \frac{1}{\pi} \left[\tan^{-1} \frac{M(k+d)}{|c|} + \tan^{-1} \frac{M(k-d)}{|c|} \right] \quad (29)$$

Equation (29) gives the probability that the relative error ξ in depth will be less than $|k|$.

Table 2 has been constructed using equation (29). The value of M used is 100, the rationale for this being the same as in section 3.1. It should be noted that this table does not contain results for image displacements less than 10 pixels in magnitude as the approximations used are not valid for small displacements.

Conclusions - From this approximate analysis we can conclude that even with large image displacements we can still get moderately high relative error in depth at all positions in most images, unless the absolute error in rotation is really quite small. We have already seen in section 3.1 that for small image displacements the relative error in depth can be very high indeed, even with very small errors in rotation.

4 Effect of Small Errors in Image Displacements on Rotational Parameters

In section 3 we found that small absolute errors in rotational parameters leads to large errors in depth. However, the question might be posed as to whether it is indeed possible to measure rotations to a very high degree of precision by any algorithm. In this section we show that it is very hard to determine rotations to the precision required in section 3 and hence it is difficult for any algorithm to determine depths.

To show this let us consider the center (0,0) of the image. Then using equation (7) we can write the rotational parameter Ω_2 as a function of the x -component of the image displacement, u

$$\Omega_2 = -u \left(\frac{1}{f} \right) \left(1 - \frac{T_3}{Z} \right) - \frac{T_1}{Z}. \quad (30)$$

To first order in the uncertainty δu , the error $\delta\Omega_2$ in Ω_2 is given by

$$|\delta\Omega_2| = \left| -\frac{1}{f} \left(1 - \frac{T_3}{Z} \right) \right| |\delta u|. \quad (31)$$

Since $(T_3/Z) \ll 1$ for almost all scenes, we can write equation (31) as

$$|\delta\Omega_2| \approx \frac{1}{f} |\delta u|. \quad (32)$$

Quantization of the image plane implies that δu is at least of order 1 pixel. In this case, for a 256×256 image with a field of view of 45° , $\delta\Omega_2$ can be calculated from equation (32) to be 0.19° . Even if we did not make the assumption that $(T_3/Z) \ll 1$ and let (T_3/Z) be $\frac{1}{4}$, which is a large translation indeed for most motion algorithms, $\delta\Omega_2$ is still 0.14° under the above conditions. In section 3 we have seen that even 0.14° absolute error in rotation can cause difficulties in depth determination both at the origin and elsewhere unless the image displacements are really very high. Hence small absolute errors in image displacements cause significant absolute errors in rotational parameters. The assumptions for our conclusion are: (a) image displacements are not measured to subpixel precisions; (b) f is of the order of hundreds of pixels or less.

In this context it should be mentioned that equation (32) suggests that the effect of uncertainties in image displacements on rotational motion parameters can be reduced by having a larger focal length. However it is

known [14] that a large field of view is necessary to facilitate the determination of motion parameters. However, for the same image resolution a large field of view means that the focal length is small. Hence it does not seem likely that this problem can be surmounted.

5 Effect of Small Displacement Errors on Depth

From sections 3 and 4 we conclude that unless the image displacements are extremely large (of the order of at least 20 pixels), it is not possible to compute depth robustly with the correspondence-based approach. Recently, some researchers have tried to avoid the issue of computing correspondence-based egomotion parameters by incorporating sophisticated navigation systems into their vehicle. However, this section shows that small absolute errors in image displacement can cause significant errors in depth even if the motion parameters are known exactly. Obviously, imprecisely known motion parameters would only increase these errors.

Taking partial derivative of equation (6) with respect to Z , we find that

$$\left| \frac{\delta \vec{l}}{\delta Z} \right| = \frac{1}{(IZ - T_3)^2} \sqrt{(\alpha I + JT_3)^2 + (\beta I + KT_3)^2}. \quad (33)$$

This implies that

$$\left| \frac{\delta Z}{Z} \right| = \frac{(IZ - T_3)^2}{Z} \frac{1}{\sqrt{(\alpha I + JT_3)^2 + (\beta I + KT_3)^2}} |\delta \vec{l}|. \quad (34)$$

Equation (34) expresses the relative error in depth as a function of depth, the motion parameters, and the uncertainty in the magnitude of the displacement vector. To understand equation (34), let us consider the most frequently attempted situation in motion - a vehicle moving straight ahead along the optical axis with no rotations⁶. Of course, this is an idealized situation and is not realizable in practice without good instrumentation. In this case the only non-vanishing motion parameter is $T_3 = 1$. Hence, equation (34) can be rewritten as

$$\hat{\xi} = \left| \frac{\delta Z}{Z} \right| = \frac{(Z - 1)^2}{Z} \frac{1}{\sqrt{x^2 + y^2}} |\delta \vec{l}| \quad (35)$$

To illustrate the relative error in depth given by equation (35), we have in Figure 4 used shading to indicate regions of the image in which the relative error $\hat{\xi}$ in depth lies in certain ranges, at a uniform depth of $Z = 10$ (i.e., the depth is ten times the total translation). We have considered that the uncertainty in the magnitude of the displacement vector is the minimum of just 1 pixel (digitization uncertainty). From the figure it can be concluded that the relative error in depth can be rather high for a very large portion of the image when the environmental points are just ten times as far away as the total translation in depth of the camera. Measurements to subpixel accuracy and higher resolution images can often reduce the error as can be observed by comparing the errors in Figure 4 and Figure 5.

⁶Although approximate motion of this type is frequently used in experimental setups this causes the image displacements to be the smallest. For depth determination it is sometimes better for the camera to move sideways rather than ahead. In some sense at least binocular stereo is better than this kind of motion for depth determination.

Table 2: Probability(Relative error in depth, $\xi \leq |k|$) using Modified Inverse Square distribution with Taylor Series approximations. $M=100$; 256×256 image; Field of View = 45°

displ. (abs val)	$\delta\Omega_2 = 0.50^\circ$					
	Relative error in depth, k					
	1%	5%	10%	25%	50%	100%
10 pix.	0.01	0.03	0.07	0.38	0.86	0.94
15 pix.	0.01	0.04	0.10	0.82	0.95	0.98
20 pix.	0.01	0.04	0.14	0.93	0.97	0.99
30 pix.	0.01	0.05	0.75	0.97	0.99	0.99
40 pix.	0.01	0.08	0.94	0.99	0.99	1.00

displ. (abs val)	$\delta\Omega_2 = 0.10^\circ$					
	Relative error in depth, k					
	1%	5%	10%	25%	50%	100%
10 pix.	0.03	0.38	0.86	0.96	0.98	0.99
15 pix.	0.04	0.82	0.95	0.98	0.99	1.00
20 pix.	0.04	0.93	0.97	0.99	0.99	1.00
30 pix.	0.05	0.97	0.99	1.00	1.00	1.00
40 pix.	0.08	0.99	0.99	1.00	1.00	1.00

displ. (abs val)	$\delta\Omega_2 = 0.05^\circ$					
	Relative error in depth, k					
	1%	5%	10%	25%	50%	100%
10 pix.	0.07	0.86	0.94	0.98	0.99	0.99
15 pix.	0.10	0.95	0.98	0.99	1.00	1.00
20 pix.	0.14	0.97	0.99	0.99	1.00	1.00
30 pix.	0.75	0.99	0.99	1.00	1.00	1.00
40 pix.	0.94	0.99	1.00	1.00	1.00	1.00

displ. (abs val)	$\delta\Omega_2 = 0.01^\circ$					
	Relative error in depth, k					
	1%	5%	10%	25%	50%	100%
10 pix.	0.86	0.98	0.99	1.00	1.00	1.00
15 pix.	0.95	0.99	1.00	1.00	1.00	1.00
20 pix.	0.97	0.99	1.00	1.00	1.00	1.00
30 pix.	0.99	1.00	1.00	1.00	1.00	1.00
40 pix.	0.99	1.00	1.00	1.00	1.00	1.00

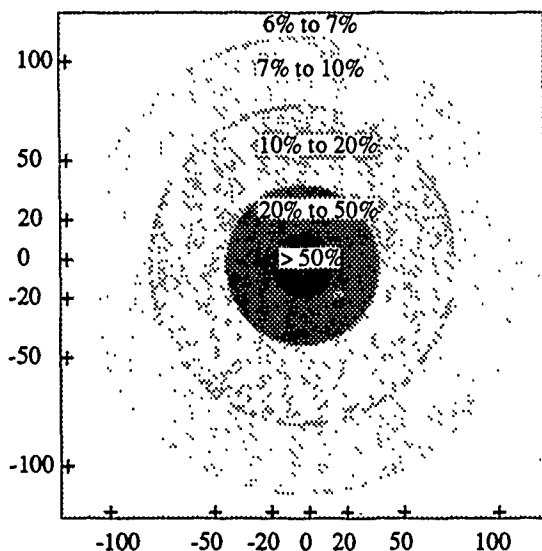


Figure 4: Relative Error in depth marked in different regions of a 256×256 image. Uncertainty in displacement vectors = 1 pixel; Field of view = 45° ; The only camera motion is translation along the optical axis and $(T_3/Z) = 0.1$.

6 Experiments on the Effect of Rotational Errors

In section 3 we theoretically studied the effect on depth recovery of small errors in rotational parameters. In order to study experimentally the effect of small perturbations of rotational parameters we generated true image displacements at various image locations with the help of the known motion parameters and known depths at these locations. Then from the image displacements we computed depth at each of the locations with the motion parameters perturbed from their true value and found the error in depth determination. That is to say we knew the motion parameters a priori so we compared the depths obtained by using the right motion parameters with those obtained by using wrong motion parameters. The wrong motion parameters were obtained by perturbing the right motion parameters. The results are

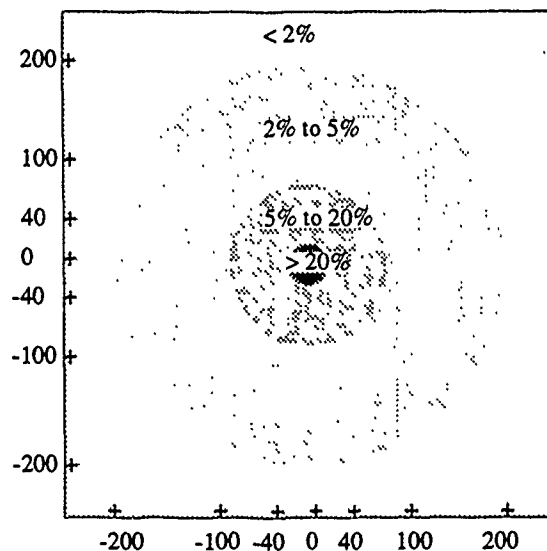


Figure 5: Relative Error in depth in different regions of a 512×512 image with uncertainty in displacement vectors being 0.5 pixel. The other parameters are the same as those for Figure 4.

shown in Table 3.

It can be seen from Table 3 that at smaller depths errors in rotational parameter have less effect. However, even when depths are in the range of 5 to 20 units and total translation is 1 unit we get over 45% error when the rotational parameters are changed by 0.1° . From the table it looks as if only rotational error of the order of 0.01° can be reasonably safely permitted even if the depths are small.

7 Conclusion

We have analyzed the important factors affecting the computation of environmental depth in correspondence-based methods. Based on all this we arrive at the following conclusions:

Effect of small rotations - Small absolute errors in rotations (even of the order of 0.1°) cause very large relative errors in depth determination. This result is

Table 3: Relative error in depth vs. Error in rotational parameters

$f = 309$ pixels ; i.e. field of view = 45°

Total translation = 1 unit

500 random points on a 256×256 grid

u, v (flow vectors) computed till 4 places after decimal

$(T_1, T_2, T_3) = (0.100, -0.100, 0.990)$

$(\Omega_1, \Omega_2, \Omega_3) = (0.300^\circ, -0.700^\circ, 0.400^\circ)$

Case 1 Depths in range 5 to 200 units

Error in $\Omega_1, \Omega_2, \Omega_3$ (deg.)			% Rel. Error	
$\delta\Omega_1$	$\delta\Omega_2$	$\delta\Omega_3$	Average	Std. Dev.
0.5000	0.5000	0.5000	240	1600
0.1000	0.1000	0.1000	1800	40000
0.0100	0.0100	0.0100	37	220
0.0010	0.0010	0.0010	3.9	33
0.0001	0.0001	0.0001	0.28	0.98
Pure Translation				
-0.3000	0.7000	-0.4000	150	520

Case 2 Depths in range 5 to 20 units

Error in $\Omega_1, \Omega_2, \Omega_3$ (deg.)			% Rel. Error	
$\delta\Omega_1$	$\delta\Omega_2$	$\delta\Omega_3$	Average	Std. Dev.
0.5000	0.5000	0.5000	150	440
0.1000	0.1000	0.1000	45	390
0.0100	0.0100	0.0100	3.4	12
0.0010	0.0010	0.0010	0.34	1.3
0.0001	0.0001	0.0001	0.02	0.12
Pure Translation				
-0.3000	0.7000	-0.4000	180	880

algorithm-independent. The errors can be reduced by increasing the magnitude of the image displacements (but this has to be balanced by the fact that this makes correspondence more difficult).

Effect of errors in image displacements on rotational parameters - Small absolute errors in image displacement cause significant absolute errors in rotational parameters. Although a larger focal length can help in reducing the error, a larger focal length implies a smaller field of view (if the image resolution is kept constant) and this makes precise determination of motion parameters difficult. Maybe two cameras of differing focal lengths are necessary for solving the problem.

Effect of errors in image displacement on depth - Even if the exact motion parameters are known, depth determination is sensitive to the error in image displacement except for very nearby points. Hence, the solution is to have large camera translations.

Our results emphasize the point that depth computations are highly sensitive to image displacement measurements. In a related sense, these results imply that efforts to incorporate hardware to find motion parameters in moving vehicles will not determine depth accurately unless these devices have a very small absolute error in rotation (with sophisticated navigation systems this may be feasible) and the image displacements are computed with very small error.

Acknowledgements - Discussions with Prof. E. M. Riseman, Prof. Allen R. Hanson, Dr. John Oliensis, Rakesh Kumar and Harpreet Sawhney proved to be of great help to the authors.

References

- [1] R. Tsai and T. S. Huang. Uniqueness and estimation of 3-d motion parameters of rigid bodies with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 13-27, January 1984.
- [2] O. Faugeras, F. Lustman, and G. Toscani. Motion and structure from motion from point and line matches. *International Conference on Computer Vision*, pages 25-34, 1987.
- [3] B. K. P. Horn. Relative orientation. *Proceedings DARPA Image Understanding Workshop*, pages 826-837, 1988.
- [4] J. W. Roach and J. K. Aggarwal. Determining the movement of objects from a sequence of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 554-562, November 1980.
- [5] Gilad Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 384-401, July 1985.
- [6] H. C. Longuet Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, pages 133-135, September 1981.
- [7] J. Q. Fang and T.S. Huang. Solving three dimensional small-rotation motion equations. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 253-258, June 1983.
- [8] J. Weng, T. S. Huang, and N. Ahuja. Motion and structure from two perspective views: Algorithms, error analysis, and error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 451-476, May 1989.
- [9] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of 3-d motion parameters and surface structures of rigid objects. In *Image Understanding 1984*, pages 135-171. Ablex Publishing Corporation, 1984.
- [10] J. L. Barron, A. D. Jepson, and J. K. Tsotsos. The sensitivity of motion and structure computations. *Proceedings Sixth National Conference on Artificial Intelligence*, pages 700-705, 1987.
- [11] M. A. Snyder. The precision of 3-d parameters in correspondence-based techniques: The case of uniform translational motion in a rigid environment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 523-528, May 1989.
- [12] R. Dutta, R. Manmatha, E. Riseman, and M. Snyder. Issues in extracting motion parameters and depth from approximate translational motion. *Proceedings Image Understanding Workshop*, Volume 2: pages 945-960, April 1988.
- [13] R. Manmatha, R. Dutta, E. Riseman, and M. Snyder. Issues in extracting motion parameters and depth from approximate translational motion. *Proceedings IEEE Workshop on Visual Motion, Irvine, Calif.*, pages 264-272, March 1989.
- [14] Gilad Adiv. Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 477-489, May 1989.
- [15] J. K. Aggarwal and N. Nandhakumar. On the computation of motion from sequences of images - a review. *Proceedings of the IEEE*, pages 917-935, August 1988.
- [16] Gilad Adiv. *Interpreting Optical Flow*. PhD thesis, University of Massachusetts at Amherst, 1985. COINS TR 85-35.
- [17] R. Dutta, R. Manmatha, L. R. Williams, and E. M. Riseman. A data set for quantitative motion analysis. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 159-164, June 1989.

COMPARATIVE RESULTS OF SOME MOTION ALGORITHMS ON REAL IMAGE SEQUENCES

Harpreet S. Sawhney

Allen R. Hanson

Computer and Information Science Department
University of Massachusetts at Amherst *

Abstract

This paper presents comparative preliminary results from the application of three motion algorithms [1,6,10] to two sequences of images obtained through a rotational motion of the camera in indoor scenes. The rotation was off-centered from the camera origin and thus structure could be derived. Two of the motion algorithms [1,6] compared represent two popular techniques in structure from motion. The third algorithm is an algorithm for reconstructing the scene from image trajectories of rotational motion [10]. All the algorithms compared use point correspondences. It is demonstrated experimentally that two-frame algorithms produce very unreliable depth estimates when both rotation and translation parallel to the image plane are present. Furthermore, even temporal integration of the two-frame estimates cannot improve the depths. This work represents an ongoing effort into developing and understanding various techniques for recovering robust scene structure from monocular motion.

1 Introduction

This paper presents preliminary results of three structure from monocular motion algorithms as applied to two sequences of images of real scenes. The algorithms compared are Adiv's [2] small-rotations based 3D estimation algorithm, Horn's relative orientation [6] algorithm and Sawhney's algorithm for estimating scene structure from image trajectories of rotational motion [10,12]. (Henceforth, we refer to these three algorithms as ADIV-3DEST, HORN-3DEST and TRAJ-3DEST, respectively). The results are compared both with hand measurements of distances and with the camera-centered coordinates of scene points obtained using Kumar's [7] pose refinement algorithm (POSE-3DEST). All of these algorithms use point correspondences as input and generate estimates of rigid motion and the 3D coordinates (up to a scale) of the imaged points.

This presentation is also intended as a progress report on results obtained from the trajectory description algorithm developed in [11,12]. In [10] a closed-form solution to the problem of extracting the circular 3D trajectory of a point given its conic image trajectory was presented. We also demonstrated the highly unreliable trajectory descriptions obtained when conics are fitted to small arcs of point tracks in the image. It was also shown that a grouping algorithm which could exploit spatio-temporal constraints across a group of point tracks could lead to

reliable combined fits and hence to accurate 3D reconstruction. As reported in [11,12], a grouping algorithm which exploits the approximately common parameters amongst a set of proximal point tracks, to incrementally group these into combined conic fits, has been developed.

The next section describes the two image sequences used and the processing involved to obtain point correspondences. Section 3 briefly describes the TRAJ-3DEST algorithm, contrasts the trajectory fits obtained from independent fits with those from the combined fit algorithm and also compares the 3D estimates obtained from the two fits. In section 4 an overview of the other two motion algorithms, ADIV-3DEST and HORN-3DEST, and the pose refinement algorithm, POSE-3DEST, is presented. Section 5 is devoted to the comparison of the four algorithms and section 6 summarizes the conclusions of this study.

2 Input Data

The two sequences used in this work were digitized with a GOULD frame grabber which outputs 512 by 484 pixel images. These were reduced further to 256 by 242 pixels for our experiments.

For the first sequence, called the *box-seq*, a rectangular chequered box was rotated around its body-axis using a cartesian robot arm and the camera looked down obliquely at it. Using a SONY B/W AVC-D1 camera, with effective FOV 24 deg. by 23 deg. (computed using the method of Lenz et al. [9]), a sequence of 20 frames was captured. The approximate angle of rotation between consecutive frames was 3.6 degrees. The range of depths in this scene was about 550 to 700 mm. Frames 1 and 10 are shown in Figs. 1 and 2, respectively.

The second sequence, called the *room-seq*, are a set of images taken inside a robotics laboratory. Objects in the scene varied in depth from 10 to 30 feet. Twenty-five frames were captured with a SONY B/W XC-77 camera mounted on a PUMA arm which in turn was mounted on a platform at one end of the room. The effective field of view (FOV) of the lens-camera and grabber systems was found to be 42 deg. by 40 deg. The arm was rotated with the axis of rotation nearly parallel to the optical axis of the camera as constrained by the configuration of the gripper. The angle of rotation between consecutive frames was 4 degrees. Frames 1 and 13 of this sequence are shown in Figs. 3 and 4, respectively.

To generate point tracks, corner-like points defined by line intersections were tracked using the line-tracking system of Williams et al. [16]. This system tracks lines obtained from the line-extraction algorithm of Boldt et al. [5] by predicting their appearances in successive frames using the displacement field output of the algorithm by Anandan [4]. Figs. 5 and 6 show a sample set of tracked lines overlaid over frame 1 lines for the

*This research has been supported by the Defense Advanced Research Projects Agency under RADC contract F30602-87-C-0140 and Army ETL contract DACA76-89-C-0017.

respective sequences. Figs. 7 and 8 depict the respective point tracks that form the input to our trajectory algorithm and the pairwise point correspondences over time among these are the input to the two-frame motion algorithms.

3 Algorithm TRAJ-3DEST

In [10] a closed-form solution to the problem of estimating the rotational trajectory of a 3D point given its conic image projection under perspective projection was developed. For reliable 3D reconstruction, it is required that stable conic fits be obtained to the discrete point correspondences over a sequence of images. In realistic situations, only a fraction of the full circular 3D trajectory can be imaged. Conic curves can be very "creative" when fitted to small sections of an arc.

We observed that the approximate commonality between the image trajectory parameters of proximal point tracks could be exploited to obtain stable combined fits to a group of point tracks. In [10] we presented results for one real sequence for which we obtained good combined fits by manually grouping a set of point tracks.

Exploiting the idea of combined trajectory grouping [11,12], an algorithm was developed which generates reliable fits to a set of proximal point tracks automatically. The two major components of this algorithm are a *grouping schedule* and a combined fit *error measure*. Given a set of proximal point tracks as input, the grouping schedule decides which track to try next in a cycle based on the covering, the gap and the overlap amongst the already grouped tracks and the new track. The idea behind this schedule is to guide the incremental algorithm to the correct solution most of the time by generating good initial guesses. The error measure is the sum of the first-order distance of a point from a conic over all the participating point tracks in a cycle. It is parameterized to make explicit the common and the independent conic parameters among a set of points. For further details refer to [11,12]. Here we present some salient results of independent fits versus combined fits on the two sequences.

Figs. 9 and 10 show the results of fitting trajectories independently to sample sets of points for the *box-seq* and *room-seq*, respectively¹. It is graphically evident from these figures that there apparently is nothing in common between the motion of the points generating these trajectories. This is amply borne out by the highly inaccurate results obtained for the 3D depth of these points (e.g. see Tables 1 and 2).

For the *box-seq*, Fig. 11 shows the results of combined fits to a sample set of point tracks obtained using the grouping algorithm described above. In contrast to the independent fits (Fig. 9), the new trajectories make the common 3D motion of the underlying points explicit by the approximate collinearity of the minor axes of various groups. Note that for this case of motion, under perspective projection, the minor axes should not be collinear globally.

Fig. 12 shows the combined fits for a sample set of tracks for the *room-seq*. There is a visually dramatic improvement in the nature of the resulting trajectories. The common axis of rotation becomes explicit by the collinearity of the minor axes of the trajectories. This makes the resulting 3D parameters very accurate. Note that for this sequence, where the rotation is nearly parallel to the image plane, our grouping constraint [12] is globally valid. The image trajectories in this case are expected to be nearly circular.

¹Note that in all the figures showing trajectories, only a subset of all the trajectories is shown for clarity.

4 The Motion and Pose Algorithms

We chose Adiv's [2] (ADIV-3DEST) and Horn's [6] (HORN-3DEST) motion algorithms for comparison for two main reasons. Firstly, these are representative of a set of techniques available for computing the rigid motion parameters between two frames using image point correspondences, followed by computation of the 3D coordinates. Secondly, the implementations of the two algorithms were easily available to us in a packaged form.

Adiv's ADIV-3DEST is representative of the set of motion algorithms which approximate the rotational transform upto the first order assuming the rotations between two frames to be small. Using Adiv's notation, the displacements are,

$$\begin{aligned} X' - X &= \alpha = -\omega_x \frac{XY}{f} + \omega_y (f + \frac{X^2}{f}) - \omega_z Y + \frac{T_x f - T_z X}{f} \\ Y' - Y &= \beta = -\omega_x (f + \frac{Y^2}{f}) + \omega_y \frac{XY}{f} + \omega_z X + \frac{T_y f - T_z Y}{f} \end{aligned}$$

where, $R = \begin{bmatrix} 1 & -\omega_x & \omega_y \\ \omega_x & 1 & -\omega_z \\ -\omega_y & \omega_z & 1 \end{bmatrix}$ is the approximate rotation matrix.

$(X, Y), (X', Y')$: Point coordinates in frames 1 and 2, .

α, β : $X-Y$ displacements over two frames.

(T_x, T_y, T_z) : The translation vector.

f : Focal length.

Adiv minimizes

$$\sum_{i=1}^N W_i * [(\alpha_{i_{meas}} - \alpha_i)^2 + (\beta_{i_{meas}} - \beta_i)^2]$$

where $\alpha_{i_{meas}}$ and $\beta_{i_{meas}}$ are the measured image displacements. He first eliminates the depth, z , from this error. Then, using a sampling on the unit sphere for translations, he computes the rotation vector for each of the sampled translation vectors. The pair of rotation and translation vectors leading to a globally minimum error is found as the solution.

Two observations can be made for Adiv's formulation. Firstly, it cannot handle large rotations. Secondly, his search method is sensitive to the quantization of the unit sphere of translations. In order to get a good performance, we ran his algorithm as three passes of a coarse-to-fine search.

Horn's [6] relative orientation algorithm represents the most widely used constraint, with slight variations, in computing motion and structure parameters from two frames of point correspondences. The constraint expresses the coplanarity of the view rays for a point in the two views with the translation vector joining the two positions. Formally,

$$f_i = b \cdot (r'_{i1} \times r_{i2}) = 0$$

where,

b : The translation unit vector in frame 2 coordinates.

r'_{i1} : Frame 1 ray for point i rotated.

r_{i2} : Frame 2 ray for point i .

Horn minimizes $\sum_i w_i f_i^2$ over all the available point correspondences in two frames with the constraint that b is a unit vector. Weight w_i is chosen as the inverse of the variance of the triple product, f_i , with respect to the variance in the measurement of the two rays.

One problem in minimizing the above measure is that the orthonormality of the rotational transformation is to be maintained throughout the optimization. All algorithms based on the essential matrix (e.g. [14,15]) derived from the coplanarity constraint do not explicitly

impose the orthonormality constraint and hence suffer with even small amounts of noise. The representation chosen for the rotational transform is crucial for good convergence.

Horn devised an elegant Gauss-Newton type technique for his error measure using quaternion representation for the rotations. The orthonormality of the rotational transform and the unit magnitude of the translation vector is maintained from iteration to iteration. Convergence to a local minimum is fast. In most cases, the local minimum with the least error is the correct solution and hence can be found by varying the starting points as suggested by Horn. However, the unweighted error measure does converge to solutions which are global minima but are wrong solutions [11,13,15].

The estimates of the ground truth were obtained by two means. For the *box-seq*, the (x, y, z) coordinates of a set of points on the three faces of the box were measured to within 1mm. Then Kumar's [7] pose estimation algorithm was run to compute the camera transformation relative to the box centered coordinate system. Pose estimates from different frames were used to reconstruct the camera centered coordinates of sample points using the resulting relative orientation. As discussed in Kumar et al. [8], given the accuracy of model measurements, the camera centered measurements were obtained to within 1% for distances of approximately 600mm. These are used as the ground truth data.

For the *room-seq*, in addition to the camera centered depth data obtained as above using pose, we also measured distances to points directly from the camera. These distances are accurate to about 0.1 feet. So for the *room-seq*, results from the motion algorithms are compared both with the direct distance measurements and estimates derived from pose.

For all the three motion algorithms, the translational scale to obtain absolute depths was provided by POSE-3DEST.

5 Comparative Results

We have performed a number of different comparisons using the algorithms discussed above. Firstly, results from independent fits versus the combined fits from TRAJ-3DEST are compared. Then we compare results from ADIV-3DEST, HORN-3DEST and TRAJ-3DEST. For the former two, depths are computed both on a two-frame basis and those averaged over a number of pairs of frames.

5.1 Box Sequence

In Table 1 we show the results of depths from individual trajectory fits versus those from the combined fits for the *box-seq* with depths derived from pose as the reference for a sample set of 12 points. The significant improvement in depths from the combined fits is evident. The percentage error in depth computation by TRAJ-3DEST are mostly within 2 % with the average at approximately 0.87 %.

In Table 3, two-frame depth estimates obtained from ADIV-3DEST and HORN-3DEST for two pairs of sample frames are compared. Forty sample points were used to compute the motion parameters which in turn were used to compute the depth estimates for twelve sample points. The first pair of frames chosen was 1-2 with an average displacement of 3.2 pixels and standard deviation of 1.3 pixels. The second set of data is from the pair 1-7 with average displacement 17.3 pixels and standard deviation of 6.7 pixels. The percent errors for 1-2 are within 10 % which go down to about 2-4 % for 1-7 as the displacements increase.

To compare the above two two-frame algorithms with our multi-frame TRAJ-3DEST, the two-frame depth

Table 1: Depths from I-Fits vs. C-Fits compared with Pose depths for the Box Sequence (IN MM.)

Pt. #	Pose Z	I-Fit Z	Err. (%)	C-Fit Z	Err. (%)
1	591.38	257.71	-56.42	588.89	-0.42
2	666.34	572.13	-14.14	665.84	-0.08
3	621.78	664.52	6.87	617.77	-0.64
4	640.66	353.75	-44.78	635.02	-0.88
5	637.68	726.36	13.91	637.71	0.01
6	647.94	370.81	-42.77	650.89	0.46
7	656.56	603.94	-8.01	661.89	0.81
8	639.99	974.83	52.32	653.80	2.16
9	709.68	675.07	-4.88	700.74	-1.26
10	614.79	56.93	-90.74	603.58	-1.82
11	602.34	527.88	-12.36	606.16	0.63
12	628.94	11.18	-98.22	636.52	1.21

estimates from six pairs of frames for each of ADIV-3DEST and HORN-3DEST were averaged. The results of this three-way comparison with pose estimates are shown in Table 4. The percentage errors for both HORN-3DEST and TRAJ-3DEST come down to within 1-2 % but for ADIV-3DEST they are slightly higher. One possible explanation for this behavior is that because Adiv approximates the rotation matrix up to the first order, his algorithm may suffer from a bias in the estimation of the motion parameters which in turn bias the depth estimates.

5.2 Room Sequence

Table 2 shows the comparison between distances from independent fits and those from combined fits for the *room-seq* for a sample set of points. Again, the tremendous improvement with the latter is evident. The average percent error is about 2.5 % with most errors well within 5 %.

Table 2: Distances from I-Fits vs. C-Fits vs. true dists. for the Room Sequence (IN FEET)

Point Num.	True Dist.	I-Fit Dist.	Error (%)	C-Fit Dist.	Error (%)
1	18.25	5.98	-67.23	18.54	1.59
2	17.94	9.73	-45.76	17.47	-2.62
3	19.59	5.60	-71.41	19.75	0.82
4	19.90	4.68	-76.48	19.75	-0.75
5	22.65	7.96	-64.86	22.84	0.84
6	29.29	5.01	-82.90	28.28	-3.44
7	25.65	6.71	-73.84	24.87	-3.04
8	26.25	13.40	-48.97	25.55	-2.67
9	14.90	3.44	-76.91	14.83	-0.47
10	14.60	5.93	-59.38	14.61	0.07
11	14.35	4.65	-67.60	15.12	5.37
12	14.70	5.86	-60.14	15.19	3.33

For the room sequence, the motion is parallel to the image plane with both rotations and translations present. It has been shown that for some restricted cases,

this case of motion leads to inherent ambiguities in rotations in depth and translations parallel to the image plane [3]. We ran Horn's algorithm on 22 points for the frame pairs 1-4 and 1-19 and Adiv's for the pair 1-4 only to limit the amount of rotation. The average T/Z ratio was about 2% for the pair 1-4 and about 10% for the pair 1-19. It was found that for both cases the errors in depth were very high. This is shown in Table 5 where we compare these two with the results from TRAJ-3DEST. The reference depths are from POSE-3DEST. The estimates from TRAJ-3DEST are well within 5%. However, the two-frame estimates are very erroneous. Furthermore, the estimates are biased well away from the true values for every pair of frames and hence, even temporal integration as in the case of *box-seq* does not improve the depths. We wish to emphasize that these results are preliminary in that a very accurate calibration of the camera center has not been performed. However, if the results of [8] can be extended to the case of motion estimation, then such large errors in depth cannot be explained by a wrong estimate of the center. There is definitely a role of the inherent ambiguities in the presence of noise in such inaccurate results. To verify this under a controlled experiment, Horn's algorithm was run on simulated data generated using a 3D model similar to that in the *room-seq* with a similar motion too. With uniform noise of 0.5 pixel added to the ideal data, the resulting errors in depth were even worse than those for the real data. We are currently studying the quantitative relationship between ambiguities in the motion parameters and the input noise for this case of motion parallel to the image plane.

6 Conclusions

We have presented preliminary results comparing our multi-frame rotational trajectory algorithm with two popular motion algorithms. For the case of motion parallel to the image plane, we demonstrated the large errors which two-frame algorithms could suffer from. However, definitive statements about this can be made only after a careful analysis and more experiments with accurately calibrated cameras. We also presented results from our grouping algorithm for rotational trajectories contrasting these with independent trajectory fits. We are continuing both analytic and experimental work to understand the performance of two-frame and multi-frame algorithms with real data to help us develop robust 3D reconstruction algorithms from monocular motion.

Acknowledgments

Many thanks to Teddy Kumar and Prof. Ed Riseman whose sustained encouragement made this work possible.

References

- [1] G. Adiv. Determining 3d motion and structure from optical flows generated by several moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(4):384-401, 1985.
- [2] G. Adiv. *Interpreting Optical Flow*. PhD thesis, University of Massachusetts at Amherst, 1985. COINS TR 85-35.
- [3] G. Adiv. Inherent ambiguities in recovering 3d information from a noisy flow field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5), 1989.
- [4] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2(3):283-310, 1989.
- [5] M. Boldt, R. Weiss, and E. Riseman. Token-based extraction of straight lines. *IEEE Transactions on Systems Man and Cybernetics*, 19(6):1581-1594, 1989.
- [6] B. K. P. Horn. Relative orientation. *International Journal of Computer Vision*, 4(1):59-78, 1990.
- [7] R. Kumar and A. Hanson. Determination of camera location and orientation. *Proceedings IEEE Wkshp. on Interpretation of 3D Scenes*, pages 52-60, 1989.
- [8] R. Kumar and A. Hanson. Pose refinement: Application to model extension and sensitivity to camera parameters. *Submitted to the DARPA Image Understanding Workshop*, 1990.
- [9] R. K. Lenz and R. Y. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3d machine vision metrology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):713-719, 1988.
- [10] H. S. Sawhney and J. Oliensis. Description and interpretation of rotational motion from images trajectories. *Proceedings DARPA Image Understanding Workshop*, pages 992-1003, 1989.
- [11] H. S. Sawhney and J. Oliensis. Description and interpretation of rotational motion from images trajectories. Technical Report COINS TR 89-90, University of Massachusetts at Amherst, 1989.
- [12] H. S. Sawhney, J. Oliensis, and A. R. Hanson. Image description and 3d interpretation from image trajectories of rotational motion. *Forthcoming TR, COINS, Univ. of Mass., Amherst*, 1990.
- [13] M. E. Spetsakis and J. Aloimonos. Optimal computing of structure from motion using point correspondences in two frames. *Proc. 2nd Intl. Conf. on Computer Vision*, pages 449-453, 1988.
- [14] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of 3d motion parameters and surface structures of rigid objects. *Image Understanding 1984*, pages 135-171, 1984.
- [15] J. Weng, N. Ahuja, and T. S. Huang. Optimal motion and structure estimation. *Proceedings Computer Vision and Pattern Recognition*, pages 144-152, 1989.
- [16] L. R. Williams and A. R. Hanson. Translating optical flow into token matches and depth from looming. *Proc. 2nd Intl. Conf. on Computer Vision*, pages 441-448, 1988.

Table 3: Two-frame Depth Estimation Comparisons

Point Num.	Pose Depth	Frames 1-2				Frames 1-7			
		Horn	Error(%)	Adiv	Error(%)	Horn	Error(%)	Adiv	Error(%)
1	591.38	553.41	-6.42	547.14	-7.48	613.88	3.80	585.36	-1.02
2	666.34	617.06	-7.40	609.60	-8.52	694.42	4.21	658.52	-1.17
3	621.78	588.97	-5.28	582.28	-6.35	648.36	4.27	619.64	-0.34
4	640.66	598.89	-6.52	593.93	-7.29	667.46	4.18	647.27	1.03
5	637.68	598.02	-6.23	591.16	-7.30	665.04	4.29	635.15	-0.40
6	647.94	608.63	-6.07	603.24	-6.90	679.23	4.83	657.39	1.46
7	656.56	614.36	-6.43	609.16	-7.22	687.45	4.70	666.35	1.49
8	639.99	600.81	-6.12	595.56	-6.94	667.96	4.37	646.77	1.06
9	709.68	666.16	-6.13	656.02	-7.56	744.84	4.95	697.09	-1.77
10	614.79	580.57	-5.57	575.09	-6.46	644.11	4.77	621.41	1.08
11	602.34	568.51	-5.62	562.74	-6.57	626.85	4.07	601.03	-0.22
12	628.94	591.09	-6.02	585.02	-6.98	655.33	4.20	627.58	-0.22

Table 4: Two-frame Depths vs. Trajectory Depths for the 12 points Compared with Pose Depths for the Box Sequence (IN MM.)

Point Num.	Pose Depth	Adiv Avgd. Depth	Error (%)	Rot. Depth	Error (%)	Horn Avgd. Depth	Error (%)
1	591.38	575.73	-2.65	588.89	-0.42	591.68	0.05
2	666.34	646.68	-2.95	665.84	-0.08	666.44	0.02
3	621.78	608.83	-2.08	617.77	-0.64	624.91	0.50
4	640.66	630.83	-1.53	635.02	-0.88	641.54	0.14
5	637.68	622.84	-2.33	637.71	0.01	639.58	0.30
6	647.94	640.01	-1.22	650.89	0.46	651.68	0.58
7	656.56	647.64	-1.36	661.89	0.81	658.75	0.33
8	639.99	630.92	-1.42	653.80	2.16	642.31	0.36
9	709.68	687.42	-3.14	700.74	-1.26	714.42	0.67
10	614.79	606.12	-1.41	603.58	-1.82	618.49	0.60
11	602.34	590.67	-1.94	606.16	0.63	604.80	0.41
12	628.94	615.43	-2.15	636.52	1.21	630.46	0.24

Table 5: Two-frame Depths vs. Trajectory Depths for 12 points compared with Pose Depths for the Room Sequence (IN FEET)

Pt. #	Pose Z	Horn Z 1-4	Err. (%)	Horn Z 1-19	Err. (%)	Rot. Z	Err. (%)	Adiv Z 1-4	Err. (%)
1	17.45	8.11	-53.5	10.08	-42.2	17.73	1.6	7.08	-59.5
2	17.47	8.31	-52.4	10.05	-42.5	17.01	-2.6	7.23	-58.6
3	19.28	8.50	-55.9	10.55	-45.3	18.43	-4.4	7.42	-61.5
4	19.27	8.42	-56.3	10.56	-45.2	19.11	-0.8	7.35	-61.8
5	22.21	8.56	-61.5	11.02	-50.4	21.74	-2.1	7.56	-66.0
6	27.61	9.10	-67.0	12.05	-56.3	26.71	-3.3	8.11	-70.6
7	24.18	8.58	-64.5	11.32	-53.2	23.48	-2.9	7.65	-68.4
8	25.07	8.89	-64.5	11.80	-52.9	24.45	-2.5	7.84	-68.7
9	14.95	8.11	-45.8	9.15	-38.8	14.71	-1.6	7.11	-52.4
10	14.27	7.76	-45.8	8.96	-37.2	14.34	0.5	6.79	-52.6
11	14.31	8.08	-43.6	8.95	-37.5	14.90	4.1	7.02	-50.9
12	14.62	7.57	-48.2	9.00	-38.4	15.11	3.4	6.53	-55.3

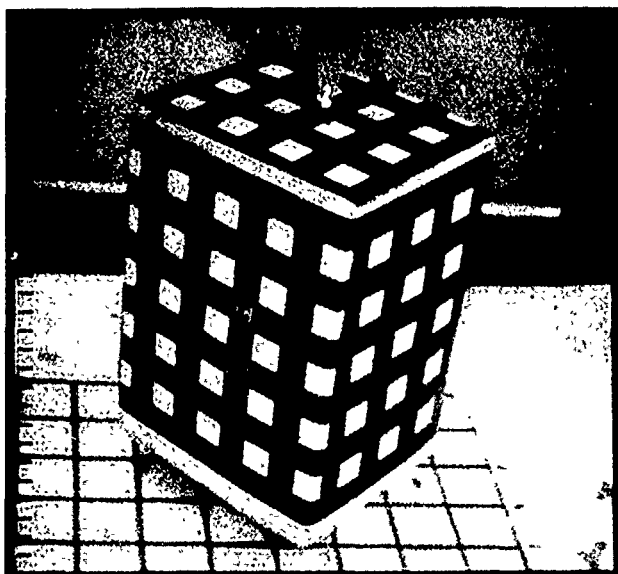


Fig. 1: Frame 1 of the Box Sequence

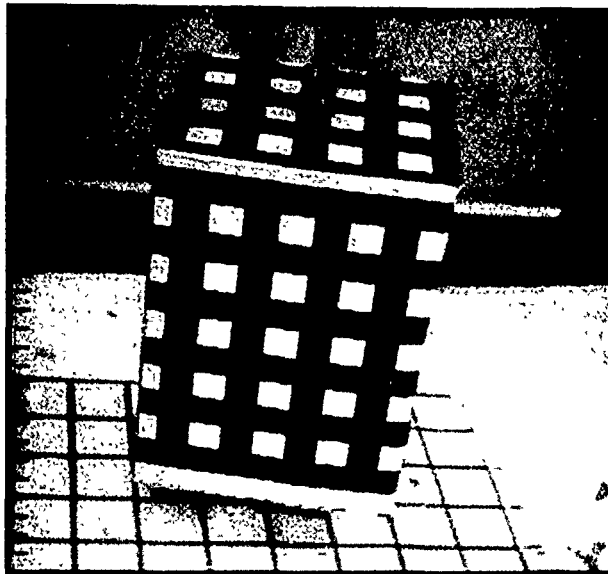


Fig. 2: Frame 10 of the Box Sequence



Fig. 3: Frame 1 of the Room Sequence

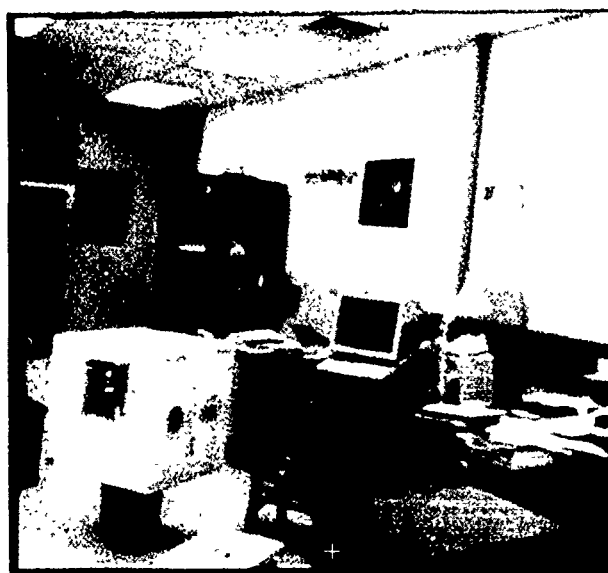


Fig. 4: Frame 13 of the Room Sequence

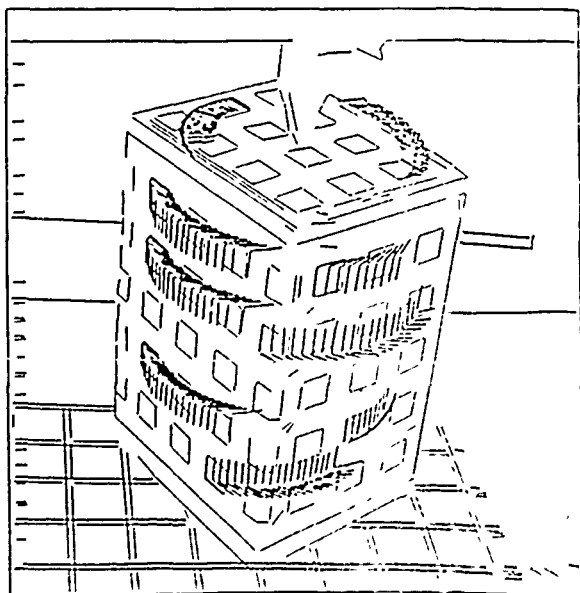


Fig. 5: Some Tracked Lines Overlaid on Frame 1 Lines (BOX)

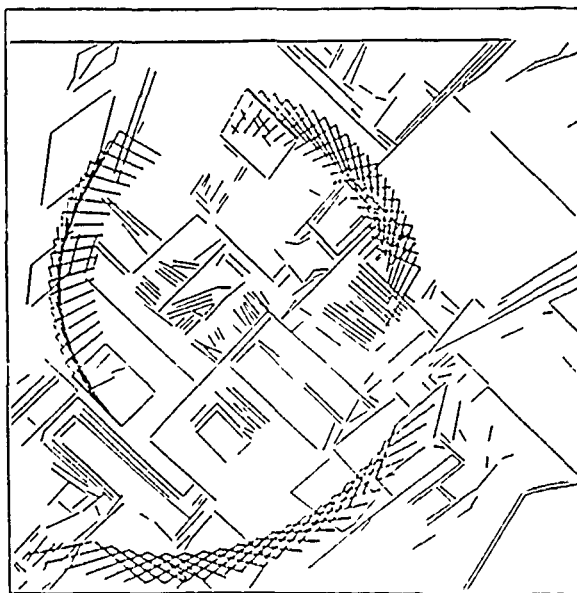


Fig. 6: Some Tracked Lines Overlaid on Frame 1 Lines (ROOM)

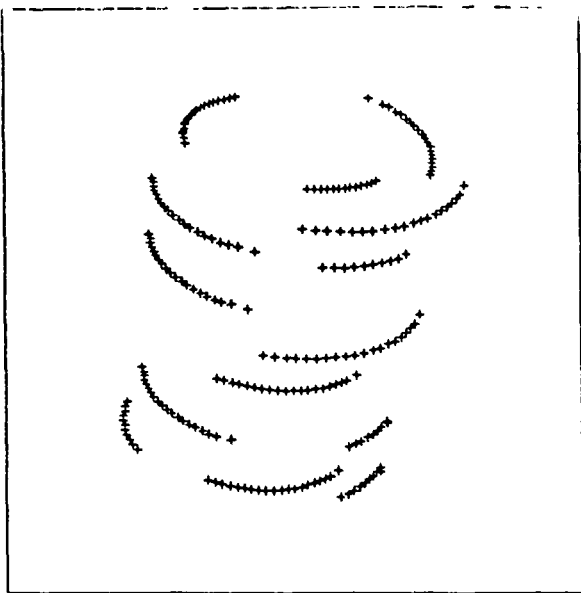


Fig. 7: Sample Set of Point Tracks(BOX)

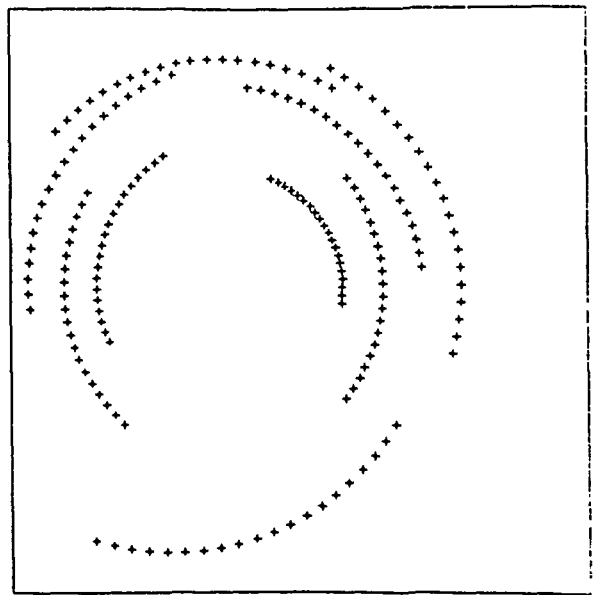


Fig. 8: Sample Set of Point Tracks(ROOM)

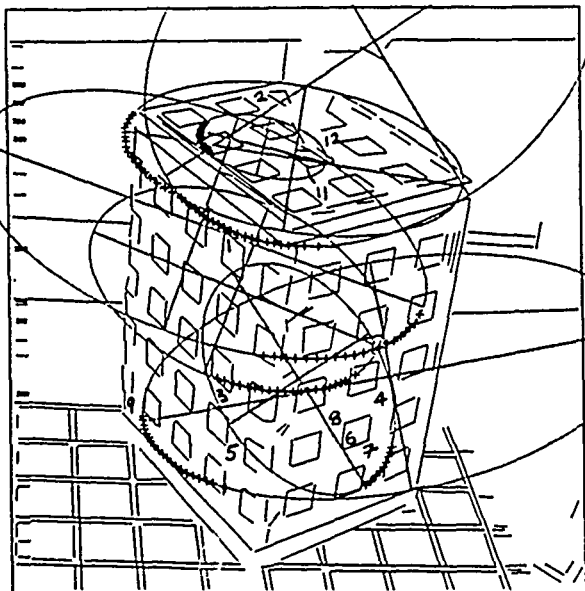


Fig. 9: Independent Fits to Box Tracks

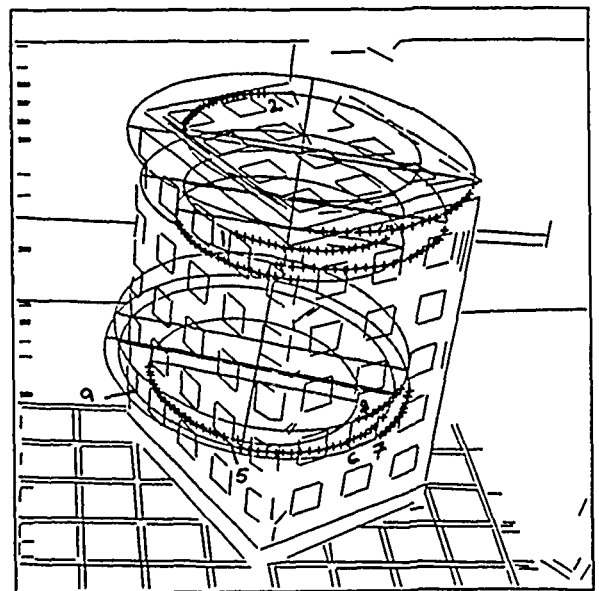


Fig. 10: Combined Fits to Box Tracks

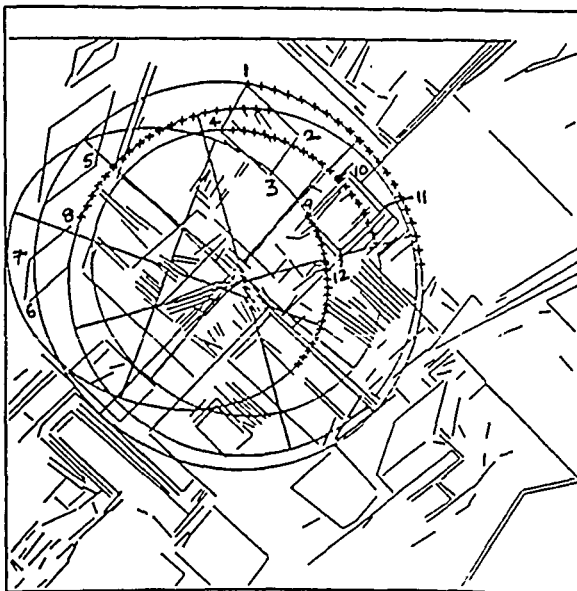


Fig. 11: Independent Fits to Room Tracks

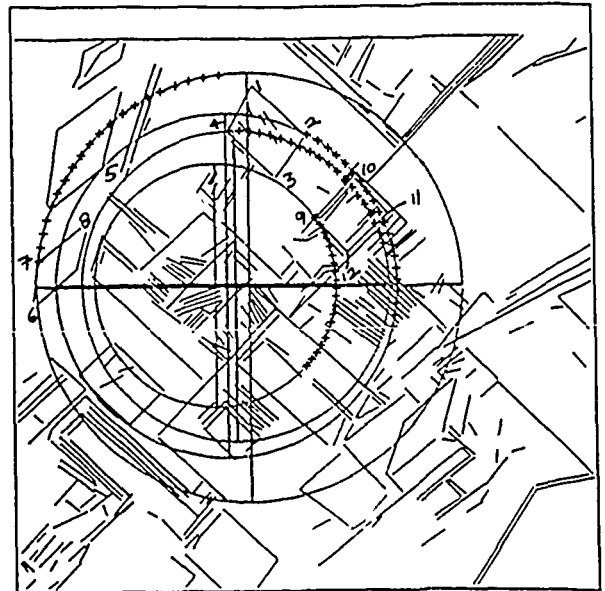


Fig. 12: Combined Fits to Room Tracks

An Estimation-Theoretic Framework for Image-Flow Computation

Ajit Singh

Vision and Robotics Group
Department of Computer Science
Columbia University, New York.

Advanced Imaging Group
Siemens Corporate Research
Princeton, New Jersey

Abstract

Image-flow is a major source of three-dimensional information. This paper describes a new framework for computing image-flow from time-varying imagery. In this framework, image-flow information is classified into two categories - conservation information and neighborhood information. Each type of information is recovered in the form of an estimate accompanied by a covariance-matrix. Image-flow is then computed by fusing the two estimates using estimation-theoretic techniques. This framework offers the following principal advantages. Firstly, it allows estimation of certain types of discontinuous flow-fields without any a-priori knowledge about the location of discontinuities. The flow-fields thus recovered are not blurred at motion-boundaries. Secondly, covariance matrices (or alternatively, confidence-measures) are associated with the estimate of image-flow at each stage of computation. The estimation-theoretic nature of the framework and its ability to provide covariance matrices make it very useful in the context of applications such as incremental estimation of scene-depth using techniques based on Kalman filtering. In this paper, an algorithm based on this framework is used to recover image-flow from two image-sequences. To illustrate an application, the image-flow estimates and their covariance matrices thus obtained are also used to recover scene-depth.

1 Introduction

Image-flow is a commonly used representation for visual-motion. It assigns to each point on the visual-field, a two-dimensional velocity vector that depicts the projection of the instantaneous three-dimensional velocity of the corresponding point in the scene. Typically, all the information that is available about a dynamic scene is an image-sequence. The image-flow field must be computed from the image-sequence. Furthermore, the process of image-flow computation must make use of *local* spatial and temporal neighborhoods. This restriction is generally imposed for reasons of computational efficiency as well as physiological plausibility.

This paper describes a new estimation-theoretic framework for image-flow computation. The principal advantages offered by this framework are as follows. (i) Covariance matrices (or alternatively, confidence-measures) are associated with the estimate of image-flow at each stage of computation. (ii) It is possible to estimate certain types of discontinuous flow-fields without any a-priori knowledge about the location of discontinuities. The flow-fields thus recovered are not blurred at motion-boundaries. (iii) Because of its estimation-theoretic nature, the framework lends itself naturally to incremental estimation of scene-depth from image-flow using techniques based on Kalman filtering. A contribution of this framework that is not discussed in this paper because of space limitations is that it serves to unify a very wide class of existing techniques for image-flow computation. The issue of unification will be discussed in a sequel paper. Before giving an overview of this framework, a brief review of the state of the art will be in order.

It is well understood [4, 15] that by using local measurements alone, the true velocity can be recovered only in those image regions that have sufficient local intensity variation, such as intensity corners, textured-regions, etc. This constitutes the well known aperture problem. Velocity must be propagated from regions of full information, such as corners etc., to regions of partial or no information. This implies that any approach to local computation of image-flow must incorporate *two* functional steps. In the first step, local information about velocity is recovered using the image-intensity distribution in small spatiotemporal neighborhoods. In the second step, the local information is propagated into neighboring regions to recover the correct image-flow. The past research is summarized below in light of these two steps. A detailed review can be seen in [2, 19]. Most of the current frameworks for image-flow computation use one of the following three basic approaches for the first step mentioned above. (i) correlation-based approach [4, 18], (ii) gradient-based approach [7, 8, 11, 15, 22] and (iii) spatiotemporal energy based approach [1, 9]. The output of the first step is in the form of initial-estimates that are updated iteratively in the second step. For the second step, the current frameworks use either a smoothness constraint [4, 10, 11, 15] or the analytical structure of image-flow [14, 21].

In the framework described here, the image-flow information available in time-varying imagery is classified into two categories - *conservation information* and *neighborhood information*. In terms of the two-step solution suggested above, conservation information

is extracted in the first-step. I call it conservation information because it is derived from the imagery by using the assumption of conservation of some image-property over time. Typically, this property is intensity [8, 11, 15], some spatiotemporal derivative of intensity [7] or intensity distribution in a small spatial neighborhood [4, 18] etc. Other choices are possible, e.g., color. Similarly, neighborhood information corresponds to the second step. I call it neighborhood information because it is derived by using the knowledge of velocity distribution in small spatial neighborhoods in the visual-field. Each type of information is recovered in the form of an estimate accompanied by a covariance-matrix. Image-flow is then computed by fusing the two estimates on the basis of their covariance-matrices.

The organization of this paper is as follows. In section 2, I show how to recover conservation information. For simplicity of presentation, I use a correlation-based approach. In the sequel paper, I show that one could use any one of the three basic approaches to recover conservation information. In section 3, I discuss the procedure for recovering neighborhood information. I also show that image-flow computation can be posed as a problem of combining conservation information and neighborhood information optimally (in a statistical sense). I present an iterative solution to this problem. I show an algorithm based on this framework in section 4 and describe the results of applying this algorithm to a variety of image sequences in section 5. In order to put this framework in context of an application, I also show the results of using the image-flow estimates to recover scene-depth using a variant of the Kalman filtering-based technique proposed by Matthies et. al [13]. Finally, I give concluding remarks in section 6.

2 Step 1: Conservation information

An implicit assumption on which most image-flow computation techniques are based is that some image-property is conserved over time. In other words, in each image of a sequence, the projection of a given moving point in the scene will have the same value of the conserved property. Factors that affect the robustness of the choice of conserved property are illumination, type of motion (rotational/translational), noise and digitization effects etc. [4, 19]. For reasons of computational simplicity, I use the Laplacian of intensity (computed by the difference-of-Gaussians operation using the masks suggested by Burt [6].) as the conserved property. I refer to the Laplacian image as just "image" for sake of brevity.

Based on the assumption of conservation, estimating image-flow using a correlation-based approach [4] amounts to an explicit search for the best match for a given pixel of an image in a search-area in subsequent images of the sequence. The extent of the search-area can be decided on the basis of a-priori knowledge about the maximum possible displacement between two images or by using a hierarchical strategy [4]. Correlation gives a *response*, i.e., a matching-strength, at each pixel in the search area. Thus, the search area can be visualized as covered with a "response-distribution". Anandan [4] had shown that using the sum-of-squared-differences (SSD) offers several computational advantages over correlation. Using SSD, which is a measure of mismatch, one obtains an "error-distribution" over the search area. The procedure for obtaining error distribution and converting it into response-distribution is discussed below.

For each pixel $\mathcal{P}(x, y)$ at location (x, y) in the first image \mathcal{I}_1 , a correlation-window \mathcal{W}_p of size $(2n + 1) \times (2n + 1)$ is formed around the pixel. A search-window \mathcal{W}_s of size $(2N + 1) \times (2N + 1)$ is established around the pixel at location (x, y) in the second image \mathcal{I}_2 . The $(2N + 1) \times (2N + 1)$ sample of error-distribution is computed using sum-of-squared-differences as:

$$\mathcal{E}_c(u, v) = \sum_{i=-n}^n \sum_{j=-n}^n (\mathcal{I}_1(x + i, y + j) - \mathcal{I}_2(x + u + i, y + v + j))^2 \quad -N \leq u, v \leq +N \quad (1)$$

The $(2N + 1) \times (2N + 1)$ sample of response-distribution is computed as follows:

$$\mathcal{R}_c(u, v) = e^{-k\mathcal{E}_c(u, v)} \quad -N \leq u, v \leq +N \quad (2)$$

The choice of an exponential function for converting error-distribution into response-distribution is based primarily on computational reasons. Firstly, it is well behaved when error approaches zero. Secondly, the response obtained with an exponential function varies continuously between zero and unity over the entire range of error.

I suggest that response-distribution be interpreted as follows. Each point in the search area is a candidate for the "true match". However, a point with a small response is less likely to be the true match, as compared to a point with a high response. Assuming that the time elapsed between two successive images is unity, each point in the search area represents a point in $n - n$ space. In estimation-theoretic terms, each of these points can be thought of as a *measurement* of the true velocity. Further, the response at the point can be thought of as a weight that reflects our faith in the measurement. One could compute an *estimate* of velocity using, for instance, a weighted-least-squares approach. Under the assumption of additive and zero-mean errors, one could also associate a covariance-matrix with this estimate. Quantitatively, the weighted-least-squares based estimate, denoted by $U_{cc} = (u_{cc}, v_{cc})$, is given by:

$$u_{cc} = \frac{\sum_u \sum_v \mathcal{R}_c(u, v) u}{\sum_u \sum_v \mathcal{R}_c(u, v)}$$

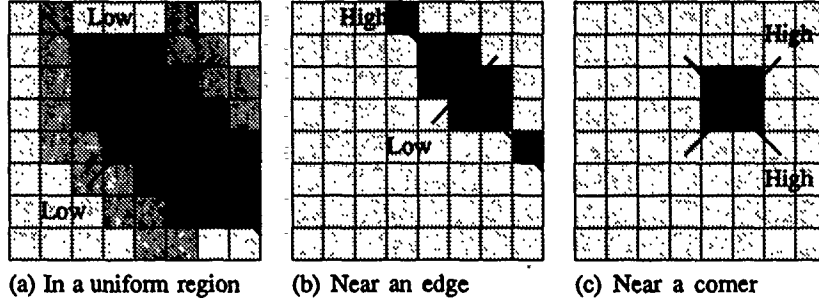


Figure 1: Response-distribution over the search-window for some representative examples - darker the pixel, higher the response. The labels "high" and "low" refer to the confidence measures associated with the eigenvectors.

$$v_{cc} = \frac{\sum_u \sum_v \mathcal{R}_c(u, v) v}{\sum_u \sum_v \mathcal{R}_c(u, v)} \quad (3)$$

and the covariance-matrix associated with this estimate is given by:

$$\mathcal{S}_{cc} = \begin{pmatrix} \frac{\sum_u \sum_v \mathcal{R}_c(u, v) (u - u_{cc})^2}{\sum_u \sum_v \mathcal{R}_c(u, v)} & \frac{\sum_u \sum_v \mathcal{R}_c(u, v) (u - u_{cc})(v - v_{cc})}{\sum_u \sum_v \mathcal{R}_c(u, v)} \\ \frac{\sum_u \sum_v \mathcal{R}_c(u, v) (u - u_{cc})(v - v_{cc})}{\sum_u \sum_v \mathcal{R}_c(u, v)} & \frac{\sum_u \sum_v \mathcal{R}_c(u, v) (v - v_{cc})^2}{\sum_u \sum_v \mathcal{R}_c(u, v)} \end{pmatrix} \quad (4)$$

where the summation is carried out over $-N \leq u, v \leq +N$. It is known [5] that reciprocals of the eigenvalues of the covariance-matrix serve as confidence-measures associated with the estimate, along the directions given by the corresponding eigenvectors. Figure 1 shows the eigenvectors and the corresponding confidence measures for some typical response distributions. Further, these eigenvectors correspond to the principal axes of response distribution. Principal axes have been used to represent velocity earlier by Scott [18].

Summarizing, there are three essential steps underlying the computation of conservation information. They are: (i) selecting the conserved quantity and deriving it from intensity imagery, (ii) computing error-distribution and response-distribution over the search-area in the velocity-space and (iii) interpreting response distribution, i.e., computing an estimate of velocity along with a covariance-matrix. The estimate, U_{cc} , can be thought of as the "initial estimate" that serves as input (along with the covariance \mathcal{S}_{cc}) to the velocity propagation procedure. As mentioned earlier, velocity propagation is accomplished using neighborhood information. Before discussing neighborhood information, the following clarification would be in order. In interpreting the response-distribution, I have assumed that it is unimodal. This assumption does get violated in the presence of texture, specially if the size of the search-window is greater than the scale of intensity variations. The weighted-least-squares approach used above "averages out" the various peaks, giving an incorrect estimate of velocity. However, since the "spread" of the distribution is large in this case (as compared to the situation where the response-distribution has a single well defined peak), the confidence associated with the estimate will be low. In essence, although the procedure for interpreting the response-distribution gives an incorrect estimate if the distribution is not unimodal, it does associate a low confidence with the (incorrect) estimate. Further, the problem of multiple peaks can be alleviated, at least partly, by using three images to compute conservation information. This is done by computing two response-distributions - one between the current image and the previous image and other between the current image and the next image - and adding the two appropriately.

3 Step 2: Neighborhood information

The objective of the second step in image-flow recovery is to propagate velocity by using neighborhood information. Assume for a moment that the velocity of each pixel in a small neighborhood around the pixel under consideration is known. One could plot these velocities as points in $u - v$ space giving a neighborhood velocity distribution. Some typical distributions are shown in figure 2. What can one say about the velocity of the central pixel (which is unknown)? Barring the case where the central pixel lies in the vicinity of a motion-boundary, it is reasonable to assume that it is "similar" to velocities of the neighboring pixels. In statistical terms, the velocity of each point in the neighborhood can be thought of as a *measurement* of the velocity of the central pixel. It is reasonable to assume that all of these measurements are not equally reliable - they must be weighted differently if used to compute an estimate of velocity of the central pixel. I weight the velocities of various pixels in the neighborhood according to their distance from the central pixel - larger the distance, smaller the weight. Specifically, I use a Gaussian mask. Based on this information, a weighted-least-squares estimate of velocity, \bar{U} , can be computed. Further, assuming additive and zero mean errors, a covariance-matrix, \mathcal{S}_n , can be associated with this estimate. The estimate and the covariance-matrix thus obtained serve as the "opinion" of the neighborhood regarding the velocity of the central pixel (as opposed to those obtained from conservation information that reflect the central pixel's own opinion).

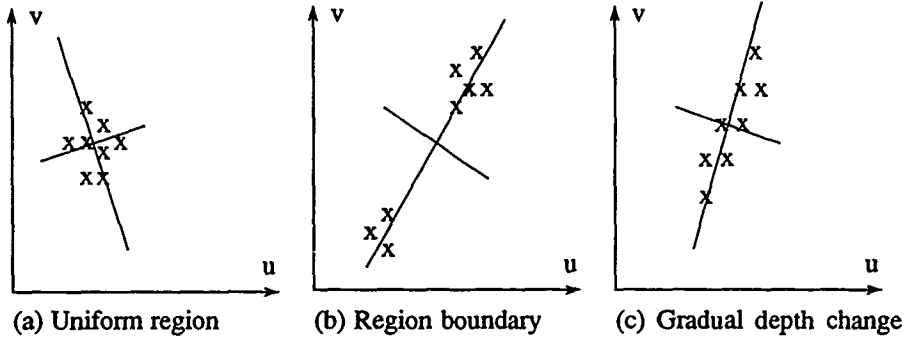


Figure 2: Velocity distribution for some representative neighborhoods.

Quantitatively, if the neighborhood size is $(2w+1) \times (2w+1)$, the velocities of these $(2w+1)^2$ pixels map to the points (u_i, v_i) in $u-v$ space (where $1 \leq i \leq (2w+1)^2$) and the weight assigned to the point (u_i, v_i) is $\mathcal{R}_n(u_i, v_i)$, the weighted-least-squares based estimate $\bar{U} = (\bar{u}, \bar{v})$, of velocity of the central pixel is given by:

$$\begin{aligned}\bar{u} &= \frac{\sum_u \sum_v \mathcal{R}_n(u, v) u}{\sum_u \sum_v \mathcal{R}_n(u, v)} \\ \bar{v} &= \frac{\sum_u \sum_v \mathcal{R}_n(u, v) v}{\sum_u \sum_v \mathcal{R}_n(u, v)}\end{aligned}\quad (5)$$

and the covariance-matrix associated with this estimate is given by:

$$S_n = \begin{pmatrix} \frac{\sum_i \mathcal{R}_n(u_i, v_i) (u_i - \bar{u})^2}{\sum_i \mathcal{R}_n(u_i, v_i)} & \frac{\sum_i \mathcal{R}_n(u_i, v_i) (u_i - \bar{u})(v_i - \bar{v})}{\sum_i \mathcal{R}_n(u_i, v_i)} \\ \frac{\sum_i \mathcal{R}_n(u_i, v_i) (u_i - \bar{u})(v_i - \bar{v})}{\sum_i \mathcal{R}_n(u_i, v_i)} & \frac{\sum_i \mathcal{R}_n(u_i, v_i) (v_i - \bar{v})^2}{\sum_i \mathcal{R}_n(u_i, v_i)} \end{pmatrix} \quad (6)$$

where the summation is carried out over $1 \leq i \leq (2w+1)^2$.

At this point, we have two estimates of velocity, U_{cc} and \bar{U} - from conservation and neighborhood information respectively, each with a covariance-matrix. An estimate of velocity that takes both conservation information and neighborhood information into account can now be computed as follows. Since this estimate is a point in $u-v$ space, its distance from \bar{U} , weighted appropriately by the corresponding covariance matrix, represents the error in satisfying neighborhood information. I refer to this error as *neighborhood error*. Similarly, the distance of this point from U_{cc} , weighted appropriately, represents the error in satisfying conservation information. I refer to this error as *conservation error*. Computing the velocity estimate, therefore, amounts to finding a point in $u-v$ space that minimizes the sum of neighborhood error and conservation error.

In quantitative terms, neighborhood error is a quadratic form commonly used in estimation theory [5] and is given by:

$$(U - \bar{U})^T S_n^{-1} (U - \bar{U}) \quad (7)$$

Similarly, conservation error is the following quadratic form:

$$(U - U_{cc})^T S_{cc}^{-1} (U - U_{cc}) \quad (8)$$

and the sum of conservation error and neighborhood error represents the squared error in the velocity estimate U . Statistically speaking, the optimal estimate of velocity is the one that minimizes the mean squared error over the visual field. That is:

$$\int \int \left[(U - \bar{U})^T S_n^{-1} (U - \bar{U}) + (U - U_{cc})^T S_{cc}^{-1} (U - U_{cc}) \right] dx dy = MINIMUM \quad (9)$$

Calculus of variations can be used to derive the optimal estimate. Let ∇_U be defined as follows:

$$\nabla_U = \begin{pmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \end{pmatrix} \quad (10)$$

The condition for minimum mean squared error can be written as:

$$\nabla_U \left[\int \int \left[(U - U_{cc})^T S_{cc}^{-1} (U - U_{cc}) + (U - \bar{U})^T S_n^{-1} (U - \bar{U}) \right] dx dy \right] = 0 \quad (11)$$

which gives [5]:

$$S_{cc}^{-1} (U - U_{cc}) + S_n^{-1} (U - \bar{U}) = 0 \quad (12)$$

In this equation, U_{cc} and S_{cc} are derived directly from the underlying intensity pattern in the image. Therefore, they are known (and fixed) for a each pixel. \bar{U} and S_n , on the other hand, are derived on the assumption that velocity of each pixel in the neighborhood is known in advance from an independent source. This assumption is invalid in practice. Hence, \bar{U} and S_n are unknown and the velocity U cannot be derived directly from equation 12. However, equation 12 is available at all the pixels in any given neighborhood in the image. If the conditions discussed below are satisfied, we essentially have a system of coupled linear equations that can be solved by an iterative technique such as Gauss-Siedel relaxation algorithm [16]. The iterative solution can be written as [16]:

$$\begin{aligned} U^{k+1} &= [S_{cc}^{-1} + S_n^{-1}]^{-1} [S_{cc}^{-1} U_{cc} + S_n^{-1} \bar{U}^k] \\ U^0 &= U_{cc} \end{aligned} \quad (13)$$

and the covariance matrix associated with the final estimate of velocity is given by $[S_{cc}^{-1} + S_n^{-1}]^{-1}$, where S_n^{-1} is computed from the final iteration. The eigenvalues of this matrix depict the confidence measures corresponding to the final estimate. The notion of final (post-propagation) covariance matrix is novel and unique to this framework. It serves several purposes. Qualitatively, it indicates what regions in the image have the most reliable image-flow estimates from the viewpoint of applicability to high-level interpretation. Quantitatively, it serves as an essential input to procedures for incremental scene-depth computation that use estimation theoretic techniques such as Kalman filtering. This is discussed in appendix A and used in depth-estimation experiments reported in the next section.

The two conditions that must be satisfied for the iterative solution to converge are discussed below. Firstly, for equation 12 to represent a system of coupled linear equations, S_n must be a constant and must be known in advance. Such is not the case here. In the current implementation, I obtain S_n from the neighborhood velocity distribution corresponding to the previous iteration. However, I have found empirically that either of the eigenvalues of S_n does not change by more than about 15% from the beginning to the end of the iterative procedure. This holds true particularly for the pixels that do not lie on a motion boundary. Secondly, for the iterative procedure to converge irrespective of the value of initial estimate U^0 , both S_{cc}^{-1} and S_n^{-1} must be positive definite. As discussed in [20], this criterion is generally satisfied in real imagery except in pathological cases such as absolutely flat regions.

So far, I have assumed that the pixel under consideration does not lie on a motion-boundary and that neighborhood velocity distribution forms a single cluster in $u-v$ space. In the following discussion, I will analyze the performance of the framework at motion-boundaries. Specifically, I will show that (i) the procedure discussed above for using neighborhood information is still justified and (ii) in absence of texture, it does a better job of preserving the step-discontinuities in the flow-field as compared to conventional smoothing-based procedures. For this purpose, recall that each of the two estimates U_{cc} and \bar{U} maps to a point in $u-v$ space. Similarly, each of the two covariance matrices S_{cc} and S_n maps to an ellipse that has its center at the respective estimate and that has its major and minor axes equal to the eigenvalues of the covariance-matrix. Therefore, each iteration amounts to finding a point in $u-v$ space that has the minimum weighted sum of squared perpendicular distances from the axes of the two ellipses - the eigenvalues serving as weights.

The behavior of this procedure in the vicinity of a motion-boundary is depicted in figure 3a. For the conservation-ellipse E_{cc} , only the major axis is shown because the minor axis will be very small in this region. In other words, all that conservation information tells (with high confidence) about the velocity of the central pixel is that it lies somewhere along the major axis of the ellipse E_{cc} . Velocities of neighboring points are also plotted from the previous iteration. Given that there is no texture in vicinity of the boundary (i.e., conservation information is reliable) and that the boundary corresponds to a step-discontinuity in the flow-field, the velocities of neighboring points form two clusters in $u-v$ space. As a result, the minor axis of the neighborhood-ellipse E_n will be very small. In other words, all that neighborhood information tells (with high confidence) about the velocity of the central pixel is that it lies somewhere along the major axis of the ellipse E_n . Since the "correct" velocity will lie in one of the two clusters, this opinion of neighborhood is correct. In other words, the iterative update procedure developed for the non-boundary pixels is justified even for pixels that lie on a motion-boundary. Furthermore, since the velocities of the neighboring pixels are derived from conservation information (at the beginning of the iterative procedure), one of the two clusters will be very close to the conservation-constraint for the central pixel. This is depicted in figure 3a. As a result, the updated velocity for the central pixel (denoted by an upper case "X" in the figure), which is given by the intersection of the two major axes, will be very close to one of the clusters. This cluster corresponds to that side of motion boundary (in the image) with which the velocity of the central pixel is more consistent. Effectively, the pixel under consideration is *binned* to the correct side of the motion boundary. For purpose of comparison, the result of conventional smoothing [4, 11] is shown in figure 3b. Clearly, the updated velocity lies somewhere in the middle of the two clusters, effectively blurring the flow-field at the boundary.

4 An Algorithm and its implementation

An algorithm based on the new framework is given below followed by the details of its implementation. The algorithm uses three images as its input. It recovers conservation information only once at the onset (steps 1 through 3) and neighborhood information once for each iteration (steps 4 through 6).

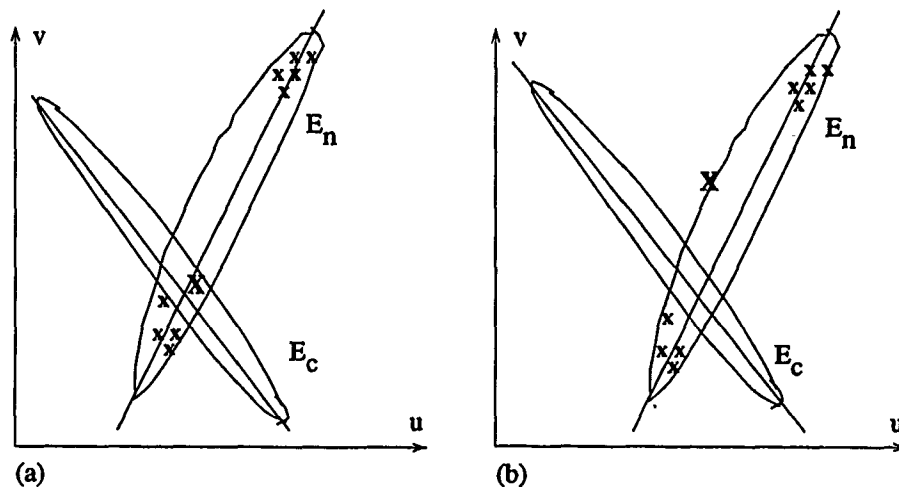


Figure 3: Performance at motion-boundaries.

Algorithm:

(1) Convolve each image with a Laplacian. (2) Form a $(2n + 1) \times (2n + 1)$ correlation-window around the pixel in consideration in the central image. Also, form a $(2N + 1) \times (2N + 1)$ search-window in around the corresponding location in the other two images. Compute the error-distributions over the two search windows and transform them to the corresponding response-distributions, \mathcal{R}_c^{-1} and \mathcal{R}_n^{-1} respectively. Finally, rotate \mathcal{R}_c^{-1} about both vertical and horizontal axes and add it to \mathcal{R}_n^{-1} to compute the resultant response distribution \mathcal{R}_c . (3) Compute the estimate U_{cc} and the covariance-matrix S_{cc} from response-distribution using equations 3 and 4 respectively. (4) Form a $(2w + 1) \times (2w + 1)$ window around the pixel in consideration. Denote each pixel by a distinct index i , where $1 \leq i \leq (2w + 1)^2$. Denote the current estimate of the velocity of i^{th} pixel by (u_i, v_i) . (For the first iteration, the velocity U_{cc} computed in step 3 can be used as the current estimate). Assign weights $\mathcal{R}_n(u_i, v_i)$ to these velocities. Compute the mean \bar{U} and the covariance-matrix S_n using equations 5 and 6 and respectively. (5) Update the velocity at the pixel under consideration using equation 13. (6) Repeat steps 4 and 5 until the change in velocity over two successive iterations is less than a threshold. (7) Compute the confidence measures associated with the final estimate of velocity as the eigenvalues of the matrix given by $S_{cc}^{-1} + S_n^{-1}$. These confidence measures are associated with the directions of maximum and minimum confidence, i.e., along the eigenvectors.

Implementation Details:

Firstly, one has to establish the parameters N , n , w and k in order to compute response-distribution. The choice of N depends on the maximum possible displacement of a pixel between two frames. If the displacement is small (of the order of one to two pixels per frame), $N = 2$ (i.e., a 5×5 search window) is appropriate. If the displacement is large, one can still use $N = 2$ along with a hierarchical search strategy [4]. The values of n and w are decided on the basis of how many neighbors should contribute their opinion in estimation of velocity of the point under consideration. Too small a neighborhood leads to noisy estimates. Too large a neighborhood tends to smooth out the estimates. Empirically, $n, w = 1$ (i.e., a 3×3 window) appears appropriate. The parameter k is essentially a normalization factor. In the implementation used here, k is chosen in such a way that the maximum response in the search-window is a fixed number (close to unity). Secondly, inversion of various matrices poses problems when one or more of the eigenvalues are zero or very small. For this reason, singular value decomposition is used for matrix-inversion. Thirdly, the choice of U_{cc} as the starting velocity for the iterative procedure is justified because it denotes the estimate that can be derived from conservation information alone. This ties well with the two-step approach to image-flow recovery - the output of the first step, U_{cc} , serves as an input to the second step. Finally, some criteria has to be established to stop the iterative update process. In the experiments reported in this paper, iteration is stopped when the magnitude of each component of velocity, when rounded to the second decimal place, does not change anywhere in the image.

5 Experiments

The experiments described in this section can be divided into two categories - qualitative and quantitative. For sake of brevity, only one experiment from each category is described. A detailed description of the objectives, methodology and results of each category of experiments is given below.

Qualitative experiments: The objective of this category is to judge the qualitative correctness of flow-fields recovered by the algorithm, specially in terms of preservation of motion-boundaries. The experiment described here uses a toy truck on a flat (and mostly dark) table. Three images are shot as the truck rolls forward. The motion is largely translational, except for in the vicinity of the wheels where it has a small rotational component. Furthermore, the motion-boundaries are expected to show up primarily as step-discontinuities in the flow-field. The images are 256×242 in resolution and the maximum image-motion is about three pixels per frame. For image-flow computations, the images are low-pass filtered and subsampled to get a resolution of 128×121 . At this level of resolution, the maximum image-flow is expected to be between 1 and 1.5 pixels per frame. In the various flow-field images that follow, the velocity vector for only every fourth pixel (in both horizontal and vertical directions) is shown for sake of clarity. Further, the magnitude of velocity is multiplied by a scale-factor of four in order to make the velocity vector clearly visible.

Figures 4 through 7 show various flow-fields and confidence measures. Figure 4a shows the central frame of the original sequence. Figures 4b and 4c show the two confidence measures associated with conservation information at each point in the visual-field. It is clear that the one of the confidence measures is high both at edges and corners of the intensity image whereas the other one is high only at corners. Figure 4d shows the initial estimate of the flow-field (i.e., the velocity U_{cc}). Figure 5 shows the flow-field after iterative velocity propagation (10 iterations), superimposed on the wire-frame of the truck. For sake of comparison, figure 6 shows the flow-field after 10 iterations of conventional smoothing [4, 11] (with the smoothing factor α set to 0.5), also superimposed on the wire-frame. For this purpose, the conservation-based estimate U_{cc} is fed into the smoothing procedure in the manner shown by Anandan [4]. A comparison of figure 5 and 6 clearly shows that the new propagation procedure does an excellent job of preserving motion boundaries. It is apparent that there is very little "bleeding" of velocity from the truck into the background in figure 5. On the other hand, there is considerable blurring of motion-boundaries in figure 6. Figures 7a and 7b show the two confidence measures after propagation. As expected, the confidence has propagated outwards from the pre-propagation high-confidence regions.

The estimation-theoretic nature of the framework and its ability to provide covariance matrices make it very useful in the context of applications such as incremental estimation of scene-depth using techniques based on Kalman filtering. One such technique was shown by Matthies, Szeliski and Kanade [13]. A variant of their scheme that uses the image-flow estimates and the covariance matrices produced by the new framework is briefly described in appendix A and is used below to recover scene-depth. For this purpose, the toy-truck experiment is repeated with the truck stationary, the camera looking from top (about 15 inches above the truck) and undergoing a one-dimensional translation in a plane perpendicular to its optical axis. Eleven frames are shot at regular intervals as the camera translates horizontally by 1.5 inches. The true depth-map (obtained with a laser range-finder) is shown in figure 8. The depth-map obtained after eleven frames is plotted in figure 9. It is apparent that the depth-estimates are very good. For sake of comparison, the depth-map obtained after eleven frames using the image-flow estimates obtained from the smoothing-based implementation described earlier is plotted in figure 10. It is apparent that the blurring of depth-discontinuities is much more prominent in figure 10. It must be emphasized that the objective of this exercise (of depth-estimation) is to put the new-framework in the context of an application, rather than to make any claims about the performance of a specific depth-estimation scheme.

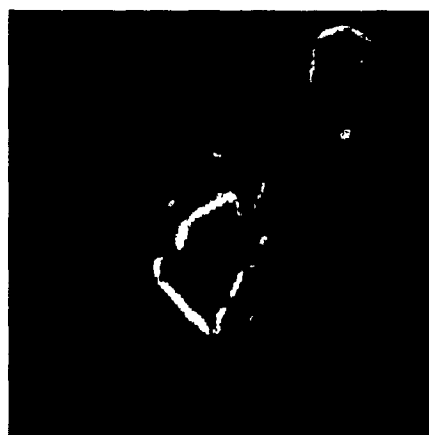
Quantitative experiments: The general objective of this category of experiments is to judge the quantitative correctness of the flow-fields. In order to accomplish this, the "ground-truth" flow-field must be known. Typically it is possible to know (or compute) the ground-truth flow-field only if (i) the motion is synthetically generated, e.g., by warping a given image in some known fashion or (ii) the camera motion and the depth of each point in the scene is exactly known. The second scenario is considered in the experiment that follows. The imagery for this experiment is selected in such a way that the flow-field does not have any discontinuities, simply because it is very difficult to come up with the ground-truth flow field in the presence of discontinuities.

Specifically, the scene is comprised of a textured poster rigidly mounted on a precision translation table. A 512×512 camera is mounted on the table as well, but its (translational) motion can be accurately controlled. The poster is placed facing the camera and slanted in such a way that (i) the optical axis is not perpendicular to the plane of the poster and (ii) the distance between the camera and the poster is very small (about 12 inches). Both these arrangements help to make the resulting flow-field interesting even when the camera is undergoing a pure translation. The camera is made to translate in a plane perpendicular to its optical axis so that the image displacement is roughly 6 pixels where the poster is closest to the camera and roughly 3 pixels where the poster is the farthest from the camera. The exact amount of camera translation as well as the distance of the lens from the rigid mount is recorded. The camera is then calibrated and its focal length is determined. The "correct" flow-field is determined using the theory developed by Waxman and Worn [21]. The images are low-pass filtered and sub-sampled to get a resolution of 128×128 using Burt's technique [6]. Both components of image-velocity at each point are divided by four to get the correct flow field corresponding to the reduced image size¹. The central image and the correct flow-field are shown in Figures 11a and 11b respectively.

¹Actually, the reduced-size imagery will correspond to image-flow that is not exactly equal to the original image-flow reduced in magnitude by a factor of four. This is because of the intensity changes that accompany low-pass filtering and subsampling. Due to lack of a quantitative characterization of these changes, I do not account for them.



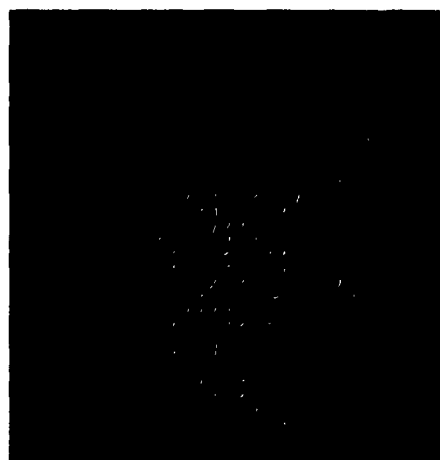
(a)



(b)



(c)



(d)

Figure 4: The toy-truck experiment: (a) central frame of the image-sequence, (b),(c) confidence measures associated with conservation information, i.e., the reciprocals of the eigenvalues of the covariance matrix $\mathcal{S}_{\mathcal{C}}$ and (d) initial estimate of velocity.

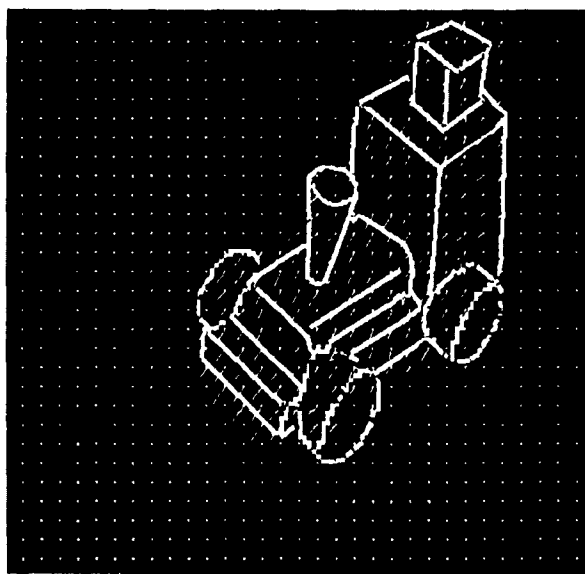


Figure 5: The toy-truck experiment: flow-field after velocity propagation, superimposed on the wire frame of the truck.

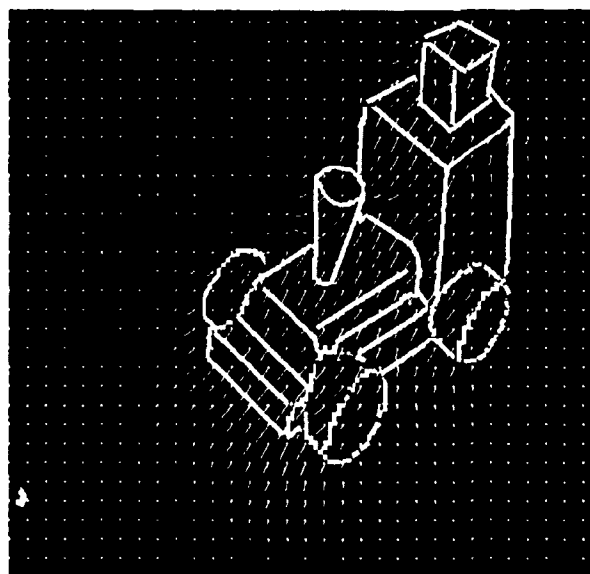


Figure 6: The toy-truck experiment: flow-field after 10 iterations of conventional smoothing, superimposed on the wire-frame

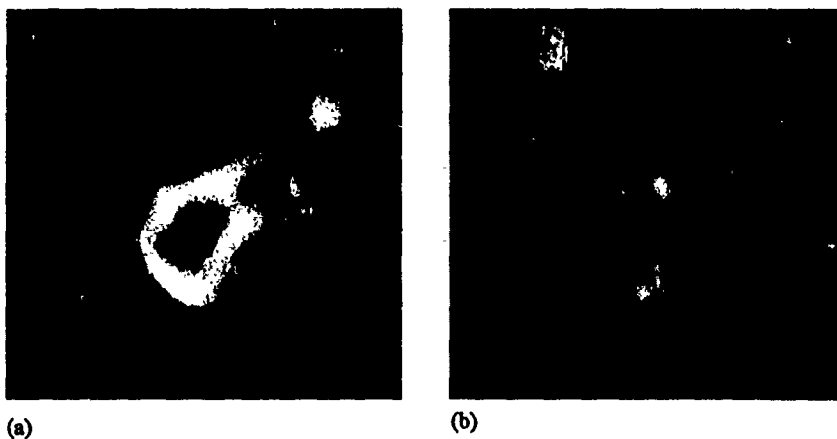


Figure 7: The toy-truck experiment: (a) and (b) confidence measures associated with the flow-field after velocity propagation.

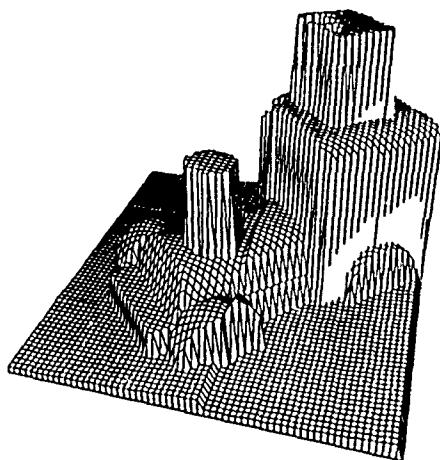


Figure 8: The toy-truck experiment: the true depth-map obtained with a laser range-finder.

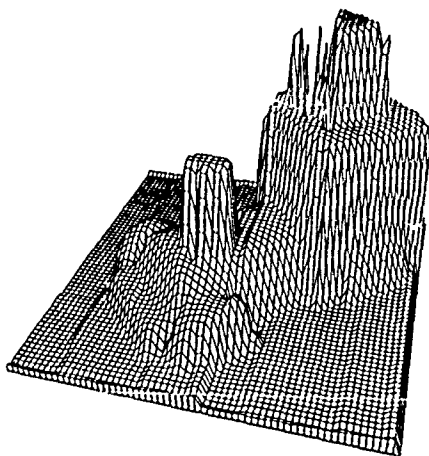


Figure 9: The toy-truck experiment: a plot of the depth-map after eleven frames using estimation-theoretic image-flow computation.

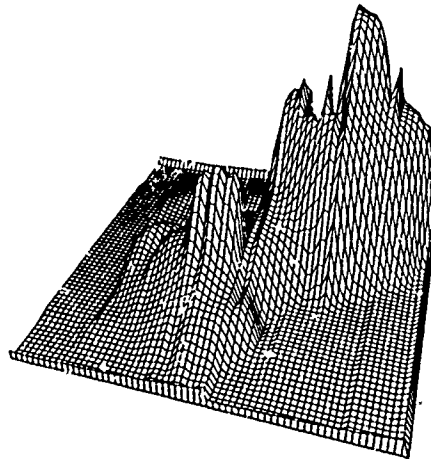


Figure 10: The toy-truck experiment: a plot of the depth-map after eleven frames using conventional smoothing-based image-flow

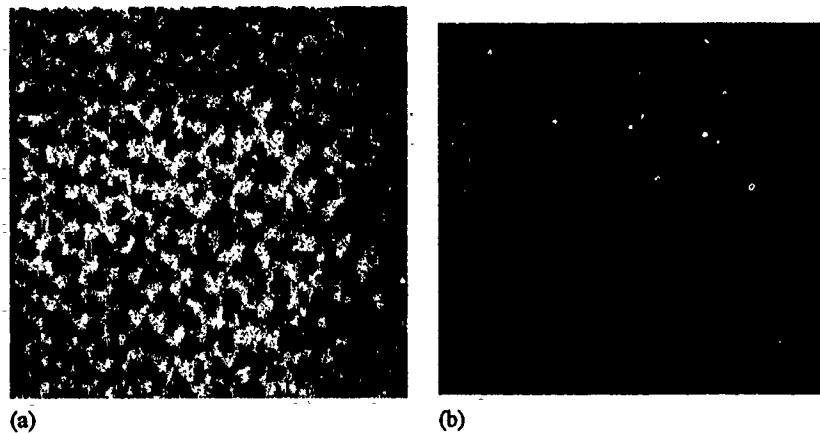


Figure 11: The poster experiment: (a) central frame of the image-sequence and (b) correct flow-field.

Two experiments are conducted, with correlation-window size set to 5×5 and 3×3 respectively. In each case, the percentage of pixels that have both components of velocity (a) within 5% (of the true value) (b) within 10% and (c) within 25%, before and after propagation (15 iterations), is determined. The results are shown in table 1. As expected, larger size of the correlation window (5×5) gives more accurate results, although reasonable results are obtained with a 3×3 correlation window also - specially after velocity-propagation.

Figures 12 through 14 show various flow-fields and confidence measures obtained with the 3×3 correlation window. Figure 12a shows one frame of the original sequence. Figures 12b and 12c show the two confidence measures associated with conservation information (i.e., the "initial" estimate of velocity) at each point in the visual-field. These confidence measures are the inverses of the small and the large eigenvalue, respectively, of the covariance matrix S_{cc} . It is apparent that the one of the confidence-measures is high both at edges and corners of the intensity image whereas the other one is high only at corners. Figure 12d shows the initial estimate of the flow-field (i.e., the velocity U_{cc}). Figure 13 shows the flow-field after iterative velocity propagation (10 iterations). It is apparent that the flow-field is qualitatively correct almost everywhere in the image, except at a few randomly placed points. The velocity-estimate at these few points is incorrect because of a very high confidence associated with a wrong initial estimate (U_{cc}). As discussed earlier, such a situation can arise in some textured regions. Figures 14a and 14b show the two confidence measures after propagation.

Once again, in order to view the image-flow estimates obtained above in the context of depth-estimation, the procedure shown in appendix A is used to recover depth-maps. Eleven frames (shot at regular intervals as the camera translate horizontally by 0.5 inch, starting from the initial configuration described before, in a plane perpendicular to its optical axis) are used. Figure 15 shows the correct depth-map. Figures 16, 17 and 18 show the depth map recovered by the procedure after three, seven and eleven frames respectively. Qualitatively, it is apparent that the depth estimates improve with time. Quantitatively, the root-mean-square error in depth (over the entire image) is 11.2%, 4.3% and 2.8% after three, seven and eleven frames respectively.

In each of the two categories, the experiments reported here have small inter-frame motion. In order to handle the cases where motion can range from very small to very large, a hierarchical version of the algorithm has been developed based on the scheme proposed by Anandan [1]. The algorithm has been tested on a wide variety of scenes (including the famous dinosaur sequence used by Anandan, where velocity is of the order of eight pixels per frame) and it works very well. The results are not included here because of space limitations.


WINDOW SIZES 	Percentage of pixels with vector error less than 5%		Percentage of pixels with vector error less than 10%		Percentage of pixels with vector error less than 25%	
	Without Prop.	With Prop.	Without Prop.	With Prop.	Without Prop.	With Prop.
5 X 5 Search 3 X 3 Correl.	53.0%	56.1%	66.4%	77.5%	71.3%	83.1%
5 X 5 Search 5 X 5 Correl.	56.2%	61.2%	68.6%	81.6%	73.2%	86.4%

Table 1. Error statistics for the poster experiment. The two rows correspond to two different sizes of the correlation window. For each row, the first and the second columns indicate the percentage of total pixels for which the error in both components of velocity is less than 5% of the correct value, before and after velocity propagation respectively. The third and the fourth columns give the corresponding percentage of pixels with error less than 10%. Finally, the fifth and the sixth columns give the corresponding percentage of pixels with error less than 25%.

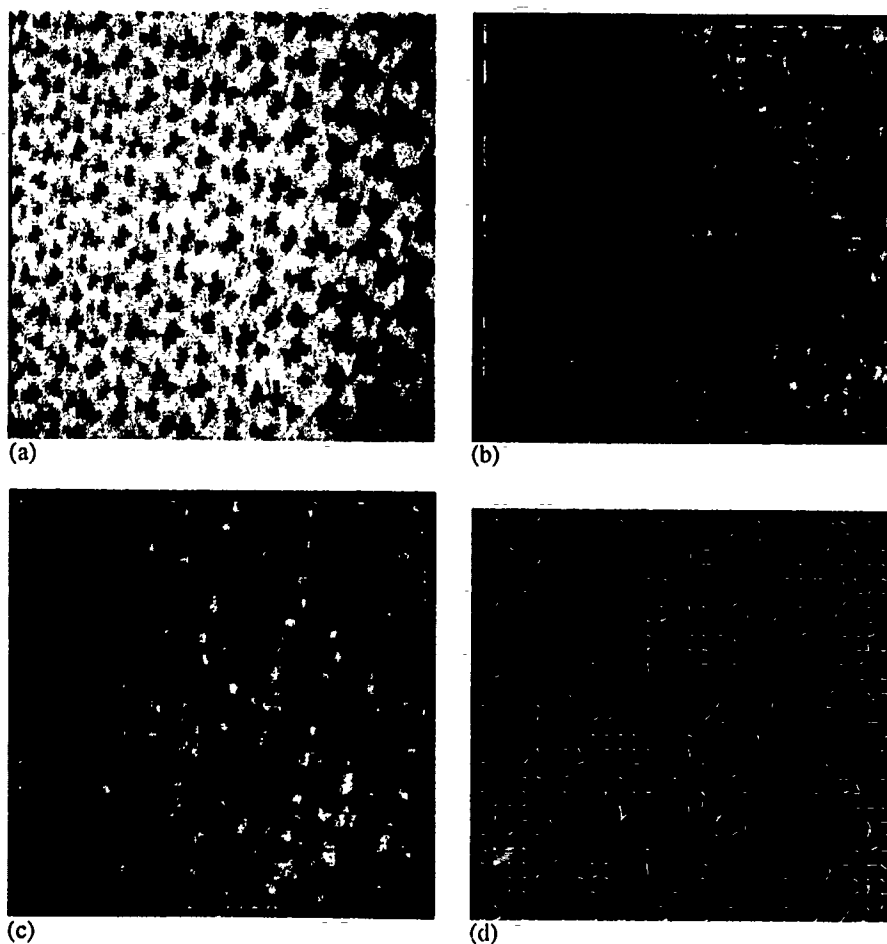


Figure 12: The poster experiment. (a) central frame of the image sequence, (b),(c) confidence measures associated with conservation information, i.e., the reciprocals of the eigenvalues of the covariance matrix S_{cc} and (d) initial estimate of velocity, i.e., U_{cc} .

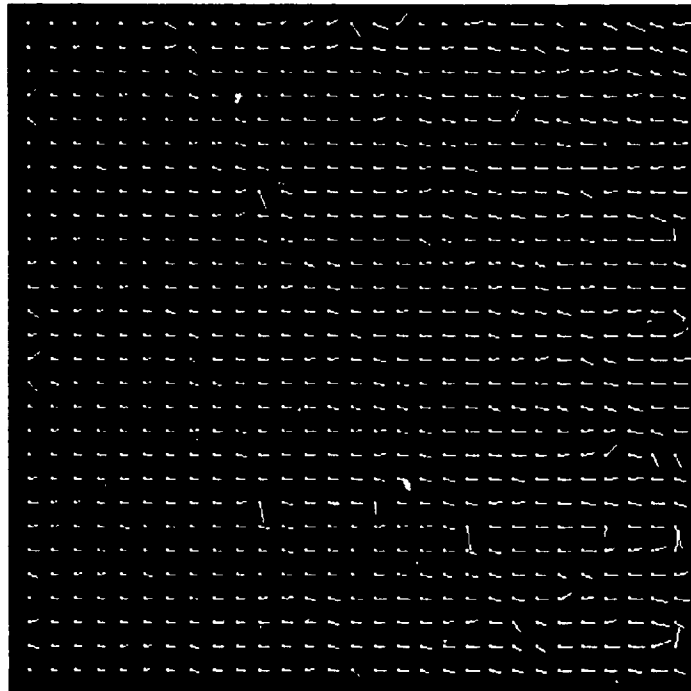
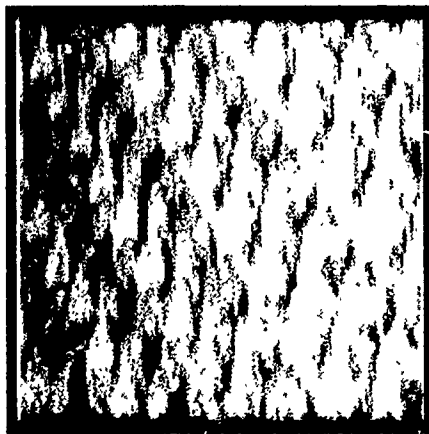
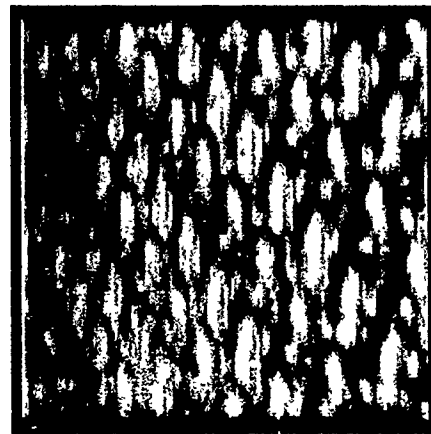


Figure 13: The poster experiment: flow-field after velocity propagation (10 iterations).



(a)



(b)

Figure 14: The poster experiment. (a) and (b) confidence measures associated with the flow-field after velocity propagation

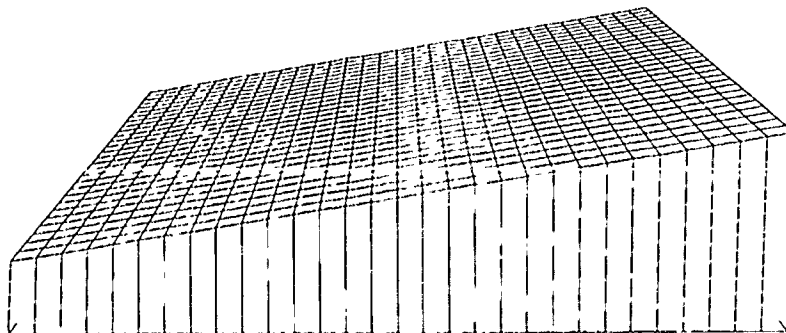


Figure 15: The poster experiment: true depth map.

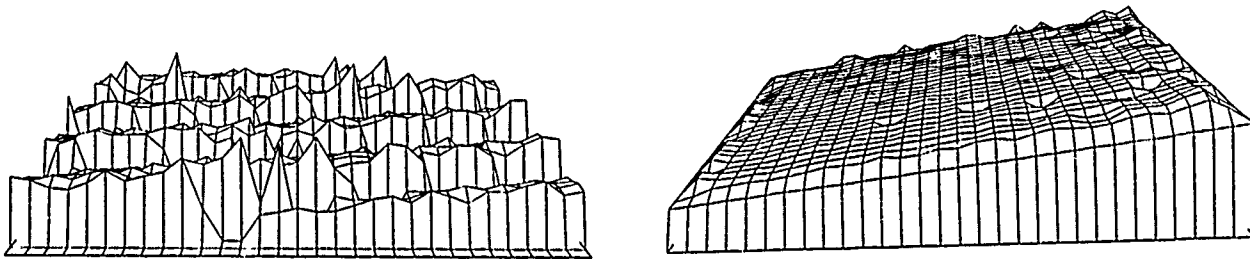


Figure 16: The poster experiment: depth map after three frames. Figure 17: The poster experiment: depth-map after seven frames.

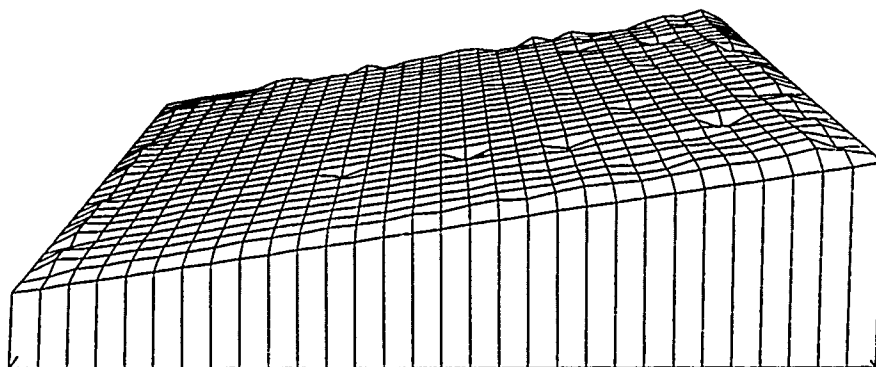


Figure 18: The poster experiment: depth-map after eleven frames.

6 Conclusion

In this paper, I have shown a new framework for recovering image-flow from time-varying imagery. This framework recognizes the fact that velocity information available in small spatiotemporal neighborhoods in the imagery is not exact - there is uncertainty associated with it. It classifies the available information into two categories - conservation information and neighborhood information - and models each one of them using techniques that are common in estimation theory. It recovers the image-flow field by performing an optimal combination of the two types of information. Some of the distinctive features of the framework are summarized below

1. It quantifies the velocity information contained in each of the two local sources - conservation and neighborhood - by an estimate and a covariance matrix. A similar approach has been used before by Anandan [4] for conservation information. However, as far as neighborhood information is concerned, this approach is novel. In essence, the current formulation accounts for the "spread" (in velocity space) of neighborhood velocities in addition to their "average" that has been used in earlier formulations [10, 11].
2. It formulates the problem of estimating image-flow as that of performing a statistical combination of velocity estimates obtained from the two sources, on the basis of their covariance-matrices. The solution to this problem is iterative and amounts to propagating velocity information from regions of low uncertainty to regions of high uncertainty.
3. Because of the statistical nature of the procedure used to represent and propagate velocity, there is an explicit notion of confidence measures associated with the velocity estimate at each pixel, both before and after propagation. The idea of pre-propagation confidence measures has been used before [4] but that of post-propagation confidence measures is novel. The experiments shown in the previous section reveal that the iterative propagation procedure used in this framework does actually enhance the confidence during each iteration. The post-propagation confidence measure reflects the reliability of the final estimate of image-flow and it can be a valuable input to a system that uses image-flow to recover three dimensional information. In the Kalman filtering-based depth estimation procedure used in this paper, the post-propagation variance (reciprocal of the confidence measure) serves as one of the inputs to the "prediction" stage.
4. The propagation procedure does a much better job of preserving the step-discontinuities in the flow field, specially in the absence of texture in the vicinity of such discontinuities, as compared to the classic smoothing based propagation procedures [4, 11]. I have demonstrated this for the toy-truck sequence and the tori sequence in the previous section. Propagation procedures used in several frameworks proposed in the recent past [3, 10, 12, 15, 17, 21] are capable of preserving motion

boundaries. However, the propagation procedure used in this framework is different from them in the following respects. (i) it gives image-flow in the entire visual-field, not just at the edges, (ii) it does not require any a-priori knowledge about the location of the boundaries, (iii) it does not assume that all intensity edges correspond to motion boundaries and vice versa, (iv) it does not use high order derivatives of the intensity function and (v) it is computationally simple.

There are several ways in which this framework can be extended and improved. Firstly, the behavior of response-distribution needs to be analyzed in greater detail, specially for the multimodal case. Secondly, in the current version of the framework, the velocity-propagation procedure utilizes only the estimate of velocity at neighboring pixels. It does not utilize the covariance-matrix associated with the estimate. It appears plausible that the knowledge of covariance matrix might assist in identifying motion discontinuities, thus making the velocity-propagation procedure even more robust at discontinuities. Finally, the formulation of optimization problem assumes that conservation-error and neighborhood error are independent. In the current implementation, however, neighborhood information is derived from conservation information. This makes the two errors dependent. An investigation of the effects of this dependence will certainly be very useful in predicting the performance of the framework with respect to any given imagery. Also, efforts could be made to ensure that the two errors are, in fact, independent.

References

- [1] E.H. Adelson and J.R. Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America.*, 2:284-299, 1985.
- [2] J.K. Aggarwal and N. Nandhakumar. On the computation of motion from sequences of images - a review. Technical Report TR-88-2-47, Computer Vision Research Center, University of Texas at Austin, 1988.
- [3] J. Aisbett. Optical flow with an intensity weighted smoothing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5:512-522, 1989.
- [4] P. Anandan. *Measuring Visual Motion from Image Sequences*. PhD thesis, COINS Department, University of Massachusetts, Amherst, 1987.
- [5] J.V. Beck and K.J. Arnold. *Parameter estimation in engineering and science*. John Wiley and Sons, 1977.
- [6] P.J. Burt. The pyramid as a structure for efficient computation. In A. Rosenfeld, editor, *Multi Resolution Image Processing and Analysis*, pages 6-37. Springer Verlag, 1984.
- [7] B.F. Buxton and H. Buxton. Computation of optic flow from the motion of edge features in image sequences. *Image and Vision Computing*, 2, 1984.
- [8] W. Enkelmann. Investigations of multigrid algorithms for estimation of optical flow fields in image sequences. *Computer Vision Graphics and Image Processing*, 43:150-177, 1988.
- [9] D. Heeger. A model for extraction of image flow. In *First International Conference on Computer Vision*, 1987.
- [10] E.C. Hildreth. *The Measurement of Visual Motion*. MIT Press, 1983.
- [11] B.K.P. Horn and B. Schunk. Determining optical flow. *Artificial Intelligence*, 17:185-203, 1981.
- [12] J. Hutchinson, K. Koch, and C. Mead. Computing motion using analog and binary resistive networks. *Computer*, pages 52-63, 1988.
- [13] L. Matthies, R. Szeliski, and T. Kanade. Kalman filter based algorithms for estimating depth from image-sequences. In *Proceedings of the 2nd International Conference on Computer Vision, Tampa, FL*, pages 199-213, 1988.
- [14] D.W. Murray and B.F. Buxton. Reconstructing the optic flow field from edge motion. an examination of two different approaches. In *First Conference on AI Applications, Denver*, 1984.
- [15] H.H. Nagel. On the estimation of dense displacement maps from image sequences. In *Proceedings of ACM Motion Workshop, Toronto*, pages 59-65, 1983.
- [16] A. Ralston and P. Rabinowitz. *A first course in numerical analysis*. McGraw-Hill Book Company, 1978.
- [17] B. Schunk. Image flow. Fundamentals and algorithms. In J.K. Martin, W.N. Aggarwal, editor, *Motion Understanding. Robot and Human Vision*, pages 23-68. Kluwer Academic Publishers, 1988.
- [18] G.L. Scott. *Local and Global Interpretation of Moving Images*. Morgan Kaufman Publishers, 1983.

- [19] A. Singh. Image-flow estimation. An analytical review. Technical Report TN-89-085, Philips Laboratories, Briarcliff Manor, New York, 1989.
- [20] M.A. Snyder. On the mathematical foundations of smoothness constraints for the determination of optical flow and for surface reconstruction. In *Proceedings of the IEEE Workshop on Visual Motion, 1989*, pages 107-115, 1989.
- [21] A.M. Waxman and K. Wahn. Contour evolution, neighborhood deformation and global image flow. Planar surfaces in motion. *International Journal of Robotics*, 4:95-108, 1985.
- [22] A.M. Waxman, J. Wu, and F. Bergholm. Convected activation profiles and measurement of visual motion. In *Proceedings of the IEEE CVPR, Ann Arbor, Michigan*, pages 717-722, 1988.

A Kalman filtering-based depth estimation from image-flow

Matthies, Szeliski and Kanade [13] had reported a Kalman filtering-based algorithm to recover dense depth maps from image-flow in the case of a stationary scene and known one-dimensional camera motion. This algorithm requires that an estimate of image-flow be produced along with its covariance for each new frame acquired (in a time-sequence) and be used to update the existing estimate of disparity (reciprocal of depth) and its variance. The principal advantage of such a scheme is that the uncertainty in depth estimates decreases with time. Matthies, et. al. had used Anandan's [4] smoothing-based algorithm to estimate image-flow and had performed error-analysis on the SSD surface to compute its variance. I have adapted their algorithm to use the framework for image-flow estimation discussed in this paper instead of Anandan's. Since this framework has an explicit covariance-matrix at each stage of computation, it fits into the Kalman filtering-based mechanism very naturally. Secondly, because of the discontinuity-preserving nature of the new framework, the discontinuities in the depth-field are better defined. This makes three-dimensional feature extraction (for interpretation of depth-fields) more reliable. Since the only modification to the original scheme of Matthies, et. al. is the way in which image-flow and its variance is estimated, the reader is referred to their original paper [13] for details of the procedure and its implementation. A block diagram of the modified scheme is shown in figure 19. The blocks 1 and 3 in this diagram depict the two steps of image-flow estimation and have been discussed in detail in this paper. The blocks 2 and 4 depict the "updating" and "prediction" steps of Kalman-filtering and are exactly the same as in [13].

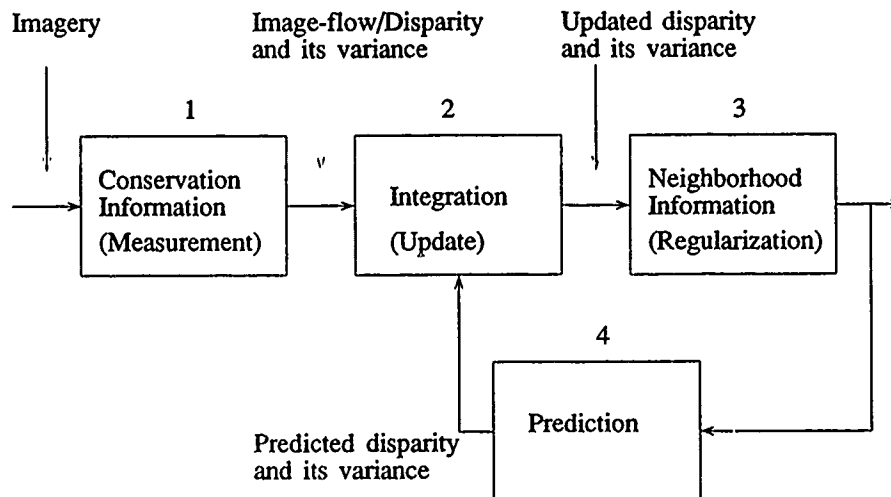


Figure 19: A block diagram of the Kalman filtering-based depth estimation scheme.

Qualitative Detection of Motion by a Moving Observer

Randal C. Nelson
nelson@cs.rochester.edu
Department of Computer Science
University of Rochester
Rochester, New York 14627

Abstract:

Two complementary methods for the detection of moving objects by a moving observer are described. The first is based on the fact that, in a rigid environment, the projected velocity at any point in the image is constrained to lie on a 1-D locus in velocity space whose parameters depend only on the observer motion. If the observer motion is known, an independently moving object can, in principle, be detected because its projected velocity is unlikely to fall on this locus. We show how this principle can be adapted to use partial information about the motion field and observer motion that can be rapidly computed from real image sequences. The second method utilizes the fact that the apparent motion of a fixed point due to smooth observer motion changes slowly, while the apparent motion of many moving objects such as animals or maneuvering vehicles may change rapidly. The motion field at a given time can thus be used to place constraints on the future motion field which, if violated, indicate the presence of an autonomously maneuvering object. In both cases, the qualitative nature of the constraints allows the methods to be used with the inexact motion information typically available from real image sequences. Implementations of the methods that run in real time on a parallel pipelined image processing system are described.

I. Introduction

The ability to rapidly detect moving objects seems to be almost universal in animals with eyes. An obvious reason is that some of the most pressing issues in the survival of an organism involve objects that move (e.g. predators, prey, and falling rocks). Robotic systems that interact with real-world environments face similar issues. They too are frequently critically concerned with objects that move. For example, an autonomous vehicle must avoid hitting people or animals that wander into its path; a surveillance system must identify intruders; and a smart weapon may pursue a moving target. A reasonable heuristic for interaction with the real world is if it is moving, you should probably pay attention. A method of detecting independent motion is thus valuable as a method for directing more sophisticated (and costly) processing to areas where it can be most effectively utilized.

For a stationary observer, a simple method of movement detection is to difference images obtained a short time apart, and mark the non-zero regions of the resulting image.

(see e.g., [Ande85]) Unfortunately, a system cannot always keep still. For a moving observer, the problem is much harder, since everything in the image may be undergoing apparent motion; and the overall pattern may be quite complex. In principle, independently moving objects can be identified through quantitative, shape-from-motion analysis [Heeg88, Burt89]. However, such approaches are generally computationally expensive, and they suffer from the fact that current shape-from-motion techniques tend to be very sensitive to the accuracy of the underlying motion data [Tsai84], often requiring a precision which is difficult to attain in practice.

A few studies have concentrated specifically on the detection of moving objects by a moving observer. Thompson [Thom90] describes some basic principles that can be used to discriminate moving objects when various aspects of the observer motion are known, but leaves open some questions about how these principles might be applied in practice. In our discussion of the constraint ray filter, we show how one of these principles can be adapted to allow the use of imprecise and partial motion information. Bhanu et al. [Bhan89] propose a method of detecting moving targets based on the identification of a fuzzy focus of expansion and a qualitative analysis of the motion of scene points. This also has some aspects in common with our proposals but it utilizes motion information derived from point correspondences, and invokes a rule-based system of qualitative reasoning, making it considerably higher level (and more expensive) than the methods described here.

We argue that the movement detection problem, can be effectively solved using a qualitative, pattern-recognition strategy. In particular, we present two complementary qualitative measures that can be used to tag motion that is inconsistent with an interpretation of global rigidity. The first method, which we term *constraint ray filtering*, makes use of knowledge about the observer's motion. It is based on the fact, noted in this context by Thompson [Thom90], that in a rigid environment, the projected 3-D velocity at any point in the image is constrained to lie on a 1-D locus in velocity space whose parameters depend only on the observer motion. Thus in principle, if the motion field and observer motion are known, an independently moving object can be detected because its projected velocity is unlikely to fall on this locus. In practice, quantitative estimates of the motion field and observer motion are both difficult and computationally expensive to obtain. We show

how the basic principle can be adapted to use partial information about the motion field and observer motion that can be rapidly computed from real image sequences. The second method makes use of qualitative knowledge about the motion of the object to be detected. It takes advantage of the fact that the apparent motion of a fixed point due to smooth observer movement changes slowly while the apparent motion of moving objects such as animals or maneuvering vehicles often changes rapidly. We term such movement *animate motion*. Such motion can be detected by using the motion field at a given time to constrain the future motion field under smooth continuation, and then looking for violations of these constraints.

The techniques presented here reflect our conviction that vision in general and motion in particular is better suited for recognition than reconstruction. This position is best clarified by examining how the two paradigms are distinguished. A major distinction is that of specificity. Reconstruction can be viewed as a general transformation of information in one form into another (presumably more useful) modality (e.g., time varying imagery into depth maps). Recognition, on the other hand, serves to identify a specific situation of interest to the system, for instance, the approach of a fly if you are a frog, or a bird if you are a fly. A reconstructed world model contains a lot of information, possibly enough to find a fly if you are a frog, but it also contains a lot of information that a frog has no interest in, and that was expensive to obtain. More specifically, a characteristic of reconstructive vision is that information is transformed without regard for its intended use, following a policy of least commitment. The usual justification is that since you never know what information you will need, you should preserve as much as possible. The disadvantage is that, since most of the information is never needed, such a policy can result in a huge amount of wasted effort, especially if attempted at higher levels. We advocate instead, what might be termed a policy of most commitment; that is, compute only what is necessary to solve the problem of interest. It might be argued that such a policy is poor science because it will never produce generalizable systems. On the contrary, we believe that the world is so structured that what is useful for one purpose will, perhaps in slightly modified form, prove useful for another. Such a statement is, of course, impossible to prove; we can only point at the history of science which is rife with examples of one structure being built on another, or at evolution, which also seems to operate in this manner.

A second distinction is the one between qualitative and quantitative methods. Reconstruction is, in essence, a quantitative procedure, and consequently dependent for its success on the numerical accuracy of the algorithms employed. This has been a problem in shape-from-x analyses in general. Recognition, on the other hand, can make use of qualitative distinctions (moving up, moving down, rotating, expanding) and relative relationships (faster, slower, in front, behind), which can be computed from much less exact information. What we feel has been overlooked is a wide variety of applications in which robustly computable motion information can be used for

identification directly, and much more efficiently, than via traditional 3-D reconstruction. The movement detection techniques presented here are one example. For others and for further discussion see [Nels88a, Nels88b, Nels89].

2. Background and Notation

2.1 Structure-from-motion

The techniques described here, although they make somewhat different use of the available information, have roots in research that has been done in the context of the structure-from-motion problem, and in methods developed to obtain local motion information from image sequences. The following is a brief review of some of the relevant terminology and results.

A camera moving within a three dimensional environment produces a time-varying image that can be characterized at any time t by a two dimensional vector-valued function f known as the *motion field*. The motion field describes the two dimensional projection of the three dimensional motion of scene points relative to the camera. Mathematically, the motion field is defined as follows. For any scene point (x, y) in the image, there corresponds at time t a three dimensional scene point (x', y', z') whose projection it is. At time $t + \Delta t$, the world point (x', y', z') projects to the image point $(x + \Delta x, y + \Delta y)$. The motion field at (x, y) at time t is given by

$$f(x, y, t) = \lim_{\Delta t \rightarrow 0} \left[\frac{\Delta x}{\Delta t}, \frac{\Delta y}{\Delta t} \right].$$

The motion field depends on the motion of the camera, the three dimensional structure of the environment, and the three dimensional motion (if any) of objects in the environment. If all these components are known, the motion field can be calculated in a relatively straightforward manner.

In the traditional approach to motion analysis, the question has been whether the process can be inverted to obtain information about camera motion and structure of the environment. This is the basis of the structure-from-motion problem. In general, the problem is ill-posed, and some assumptions (most commonly involving rigidity and surface continuity) must be made in order to attempt a solution. Various approaches to regularizing the problem have led to large body of literature on the subject. The problem has been addressed both in terms of isolated data points [Ullm79, Tsai81, Long81, Tsai84], and in terms of a full motion field and its derivatives [Praz81, Boll87, Waxm87], and a number of solutions have been described. Most of these studies, however, have started with the assumption that detailed and accurate information, either in the form of point correspondences or dense motion fields, is available. Unfortunately, the solutions to the equations are frequently inordinately sensitive to small errors in the motion field, which has made them difficult to apply in practice.

From the other direction, a number of methods have been proposed for obtaining motion information from image sequences. There have been two main approaches.

One uses matching methods similar to those employed in stereo vision to identify corresponding points [Moro79, Barn80]. This process is well known to be difficult since features may change from one image to the next, and even appear and disappear completely. The other attempts to determine the motion field from local computations of the spatial and temporal derivatives of the gray scale image. The first derivative methods originally proposed by [Horn81] must deal with what is known as the *aperture problem*, which refers to the fact that only the component of image translation parallel to the gradient can be recovered locally from first order differential information. As both methods must deal with incomplete and inaccurate information, much research has concentrated on techniques for cleaning the data and filling in the gaps. [Horn81, Schu84, Anan85, Nage86]. Well known variations of the basic methods include using higher order derivatives [Nage83, Uras88], spatio-temporal energy methods [Heeg87], Fourier methods based on phase correlation [Burt89], and direct correlation of image patches [Barn80, Litt88].

On the whole, despite a great deal of effort expended in devising motion invariants, regularization methods, and matching techniques, neither correspondence nor differential field methods have yielded data sufficiently accurate to allow the theoretical structure-from-motion results to be reliably applied. Adiv [Adiv85] argues that inherent near ambiguities in the 3-D structure-from-motion problem may make unfeasible the extraction of information sufficiently precise to allow uniform application of the theoretical solutions. Verri and Poggio [Verri87] make essentially the same point, arguing that the disagreement between the motion field and the optical flow makes the computation of sufficiently accurate quantitative values impractical. An alternative is to devise qualitative strategies that can make use of partial or inaccurate motion field information [Thom86, Nels88a, Nels89]. The movement detection strategies described here represent one such application.

2.2 Notation: spherical images and the local frame.

We consider the image formed by spherical projection of the environment onto a sphere of radius p termed the *image sphere*. The use of spherical projection makes all points in the image geometrically equivalent with respect to the observer, which considerably simplifies some of the analyses. In particular, we can define local coordinate systems with respect to an arbitrary image point p . The locations of points in the environment are expressed in terms of coordinates (X,Y,Z) where $(0,0,0)$ coincides with the center of projection, and the positive Z axis passes through p . Image positions in the neighborhood of p are expressed in terms of coordinates (x,y) where $(0,0)$ coincides with p and the x and y axes with the local projection of the X and Y axes respectively. This is permissible because the image sphere is locally Euclidean. The Euclidean neighborhood of p will be referred to as the *local projective plane*. Since all points in the image are geometrically equivalent under spherical projection, we can notationally simplify much of our local analysis by carrying it out in terms of these local coordinate systems.

Ordinary cameras do not utilize spherical projection, but if the field of view is not too wide, the approximation is reasonably close. Since the distortion is purely geometric in origin, it could be corrected should it prove to be a problem for any particular camera. In experiments we performed using a camera with a field of view of approximately 20×30 degrees, no correction was necessary in order to obtain good results.

3. Movement Detection via Constraint Ray Filtering

3.1 Theoretical basis

The first method of detecting an independently moving object is based on the observation that the projected motion at any point on the image sphere is constrained to lie on a half line (ray) in local velocity space whose parameters depend only on the observer motion and the location of the image point. In other words, despite the fact that objects at different depths typically display different apparent motion, the possibilities are constrained to a one dimensional locus in a two dimensional space. On the other hand, the projected motion for an independently moving object is unconstrained and is unlikely to fall on this locus. Thus testing the motion field to determine whether it is consistent with the local constraint ray provides a means of detecting non-rigid motion. The basic nature of the constraint is fairly well known, and it has been discussed as a means of movement detection [Thom90]; however its adaptation to partial and inexact motion information for use in a fast, practical system does not appear to have been much developed.

As a simple motivating example, consider the case of an observer translating to the right while looking straight ahead. The apparent motion of imaged objects rigidly attached to the world is horizontal and to the left with magnitude inversely proportional to the distance to the projecting world point. Any point of the motion field that contains a vertical component must thus arise from an independently moving object. Constraint ray filtering is a generalization of this idea.

To see how the constraint ray arises consider the local projective plane centered at point p on the image sphere. The projected velocity at point p is expressed by the vector (u,v) where u and v are the apparent (angular) velocities parallel to the local x and y axes respectively. The rotational motion of the observer can be decomposed into components ω_x , ω_y , and ω_z parallel to the local X , Y , and Z axes. The projected velocity due to this rotation is given by $V_\omega = (\omega_x, \omega_y)$ independent of the distance to the world point projecting to p . Similarly, the translational motion of the observer can be decomposed into components v_x , v_y , and v_z , again parallel to the local axes. In this case, the projected velocity is given by $V_t = (v_x/Z, v_y/Z)$ where Z is the distance from the origin to the world point that projects to p . The net projected velocity is the sum of the two pieces is

$$V = V_\omega + V_t = V_\omega + \frac{1}{Z} V_{XY}$$

where V_{XY} is (v_x, v_y) . For a given observer motion, both V_ω

and V_{XY} are uniquely determined, and $1/Z$ runs from 0 to $+\infty$. The possible values for V thus lie on a ray in velocity space parallel to V_{XY} and with endpoint V_ω (Figure 1).

3.2 Practical considerations

As noted in Section 2, many methods for approximating the motion field utilize low-level computations that provide only the component of the field parallel to the local image gradient. These components are then combined by various methods to obtain an approximation to the complete motion field. Since such methods are often computationally expensive it is worthwhile to examine the constraints that can be placed on the gradient parallel component and to consider whether it alone might provide information which could be used to identify independently moving objects. Consider a point x in velocity space representing the value of the motion field somewhere in the image. The gradient parallel component of the motion field can be represented by the vector describing the projection of x onto the line that is parallel to the gradient and that passes through the origin. Recall from elementary geometry that chords drawn from diametric points on a circle to a third point on the circle meet at a right angle. Thus all points on the circle whose diameter is the line segment \overline{ox} represent possible gradient parallel projections of x (Figure 2). Conversely, all lines passing through the origin intersect the circle at a point representing the projection of x onto them. Hence this circle represents all the possible gradient parallel components consistent with the motion vector. Suppose that the $1/Z$ lies between 0 and α . Then the possible image motions lie on the line segment with endpoints V_ω and $V_\omega + \alpha V_{XY}$. Each

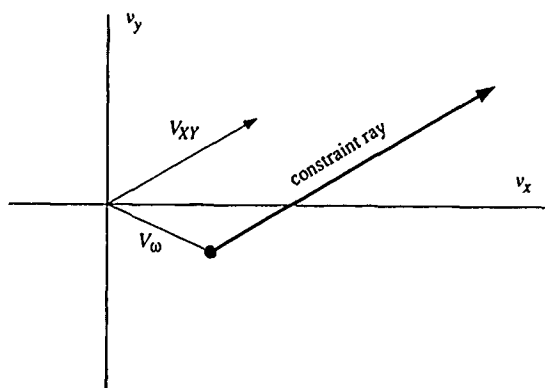


Figure 1: The constraint ray generated by the parametric equation $V = V_\omega + V_{XY}/Z$ for Z ranging from zero to infinity. Intuitively, the ray is generated by adding all positive multiples of the vector V_{XY} , whose direction is determined by the observer's translation, to the constant vector V_ω , which is produced by the observer's rotation. The axes v_x and v_y represent the components of the image velocity.

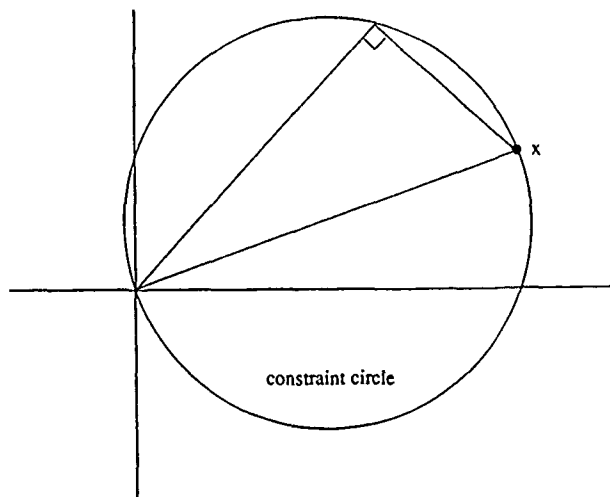


Figure 2: The constraint circle representing all possible projections of the point x onto lines passing through the origin, consequently all possible values for the gradient parallel flow component.

point on the line segment generates a circle as described above. The union of all these circles thus represents a constraint on the gradient parallel component of the motion field. It is easily seen that all these circles pass through both the origin and the projection of the origin on the constraint ray (or its extension). Thus the constraint region can be determined from the circles generated by the segment endpoints. In particular, the constraint region is the union of the two solid circles less their intersection (i.e. their exclusive OR). Figure 3 shows the partitions for several situations. In the limiting case of $\alpha = \infty$ ($Z = 0$) the partitions are formed by the intersection of a circle and a half space.

The fraction of the plane representing gradient parallel components consistent with a rigid environment is frequently sufficiently small that an independently moving object has a good chance of generating gradient parallel components that fall outside of this region. This is particularly true if α can be bounded (e.g. by knowing that the observer is at least a certain distance from the nearest object). Thus a movement detector can be constructed that utilizes local results of a differential motion computation. Such a detector would exhibit more false negatives than one utilizing the complete motion field, but on the other hand, the approximation of the gradient parallel component is far less computationally expensive.

The next issue is determining the motion of the observer. It was assumed in the above analysis that this motion was known. In some situations, such information might be available from external sources, for instance, from inertial sensors or from explicit knowledge of the observer

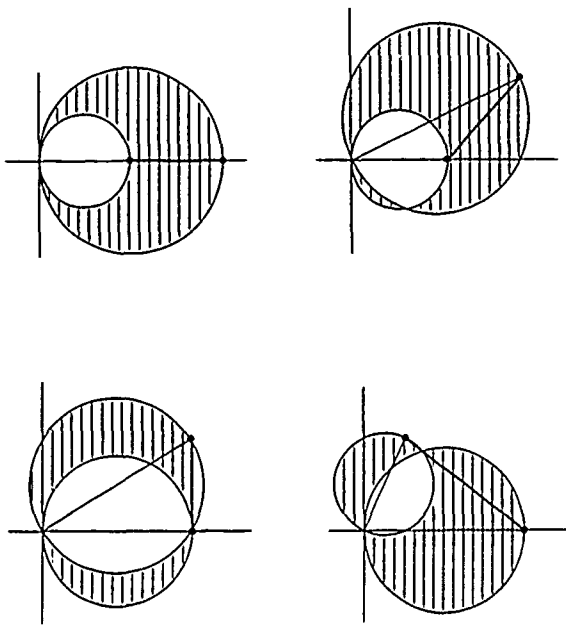


Figure 3: Constraint regions generated by segments of the constraint ray in several situations. The shaded areas represent the portion of velocity space in which the gradient parallel component of the motion field is constrained to lie when a bound can be placed on how close objects may be to the observer. If no such bound can be justified, the constraint ray extends to infinity, and the bounding disk associated with the far end becomes a half plane.

motion (stationary robots). For many applications though, it is desirable to have a self-contained system that does not rely on outside sources of information. Unfortunately, determining the observer motion from an image sequence is, in the general case, tantamount to solving the structure from motion problem for static scenes, which can be hard to do. Assuming such information to be available might thus seem to be begging the question concerning the hardest part of movement detection.

It turns out, however, that the technique can be used in practical cases without the ability to determine observer motion exactly. Two facts make this possible. First, recall that a major difficulty of solving the ego-motion from motion problem arises from the fact that, in certain situations, the motion field arising from rotation and translation can be very similar, while the implications of each about the 3-D structure of the world are very different. In our case, however, the constraints arising from such similar fields are similar. Thus, unlike the case for structure from motion, it does not much matter if the two sources are confused. Second, for many practical problems, the system spends most of its time executing only a few types of motion. For

instance, in the case of camera mounted in a car and stabilized against high frequency rotational jitter (much as the eyes of most mammals are stabilized by the reflexive VOR system), the motion is primarily straight ahead with slow rotations about the axes perpendicular to the forward motion as the car goes around corners and up and down hills. This suggests that the necessary information about observer motion can be obtained by matching against a relatively small set of prototype motion fields. The following are examples of canonical motion fields that would be useful when utilizing images having angular extent small enough so that the distortion produced in projecting the spherical image onto a plane is relatively small. (i.e. $< \text{about } 40 \times 40 \text{ degrees}$).

1. Field that is all approximately in the same direction: This corresponds either to rotation about an axis approximately perpendicular to the direction of the gaze, translation roughly perpendicular to the direction of gaze, or a combination of rotation with a translation such that the directions of the flows align. Such motions frequently arise in systems navigating on approximately flat terrain. The constraints effectively exclude motion with a component parallel to the dominant direction but of opposite sign, or with a significant perpendicular component.
2. Field having a focus of expansion in center of image: This corresponds to translation in the direction of the gaze. Here the constraints exclude motion towards the origin or having a significant tangential component.
3. Field with a distinct focus of expansion anywhere in the image (2 above is a special case). This corresponds to pure translation, or to movement while fixating on a distant point.
4. Field having an expanding periphery with uniform components perpendicular to lines passing through the image origin. This corresponds to straight ahead motion with slow rotation about a perpendicular axis. The rotation can be obtained from the motion field components normal to a perpendicular pair of lines through the origin (e.g. the local x and y axes in the image) and used to set the constraints.
5. Field that is all in one of two directions 180 degrees opposed. This results from fixating an object in the scene while moving in a direction roughly perpendicular (e.g. ± 20 degrees) to the direction of gaze. Fixation on a point at infinity or the nearest point in the image will produce a field fitting the criteria for case 1. The constraints exclude motion with a significant perpendicular component.

These cases all have robust signatures that allow them to be identified from relatively sparse information using simple pattern classification techniques (e.g. nearest-neighbor methods) They also span a wide range of motions, covering many of the situations that occur in practice in moving systems.

There are a few situations that will cause trouble. The most common arises when an isolated nearby object appears in front of a distant background. Without external information about either the observer's movement or the distance to the object, there is no way to determine whether the object is stationary or undergoing uniform motion. This would be a problem with any movement detection system. Another, which might be called the moving moon illusion, results from an isolated distant object and a strong, flat foreground. In this situation, it is possible to fixate the foreground and interpret it as distant, whereupon the distant "moon" appears to move.

3.3 Implementation

We have implemented a movement detector based on the above principles that operates in real time ($\sim 1s$ latency), and robustly detects independent motion in a wide variety of situations. The first step is the computation of local motion information. We use a differential method similar to the first step of the Horn and Schunck algorithm [Horn81], dividing the temporal derivative by the gradient magnitude to obtain an estimate of the gradient parallel component of the motion field. This operation is performed on a 512×512 video signal at 30 hertz using a collection of Datacube Maxvideo image processing boards, and provides usable values for angular velocities between about 20 and 200 pixels per second (see Appendix A). This array is then subsampled to 64×64 and downloaded to a Sun 360. A Hough transform technique is used to rapidly compute a coarse representation (here a 4×4 array quantized to one of 8 directions) of the true motion field from the gradient parallel values. This is normalized to form a feature vector which is then compared against a stored library of canonical motion fields in order to determine which of the known types of motion the observer is making. Currently the system recognizes motions in classes 1 and 2 described above. The canonical field is used to generate a filter image which specifies, for every point in the image, the range of gradient parallel components consistent with the presumed motion. The filter image is then compared with the measured estimates of the gradient parallel component, and regions exhibiting inconsistent motion are marked as potentially containing independently moving objects. By using bit encodings to coarsely represent the motion field, an update rate of about 10 Hertz was achieved.

The system has been tested using the Rochester Robot which consists of a two-eyed (we used only one), three degree of freedom head attached to a six degree of freedom robot arm, to provide observer motion [Brown88a]. For the cameras we used, having a field of view approximately 20×30 degrees, a surprisingly large range of what might be considered "natural" movements produce image motion which matches case 1 above (all approximately in the same direction). This included rotations about and translations along not only axes parallel to the picture plane, but almost any axis which did not actually intersect the image. Even with the sacrifices in resolution and accuracy made in the interest of achieving real-time performance, the system proved quite successful both at

detecting independently moving objects, and ignoring the apparent movement due to its own motion. The expected exception occurred when the independent movement was in the same direction and near the same velocity as the apparent motion, corresponding to landing on the constraint ray. Figure 4 shows the detector's response to a person walking across the field of view as the observer rotates and translates so that the entire scene appears to moving upward. The magnitude of the motion field due to the camera motion is of the same order as that due to the walker. Everything in the image is moving, yet the system reliably

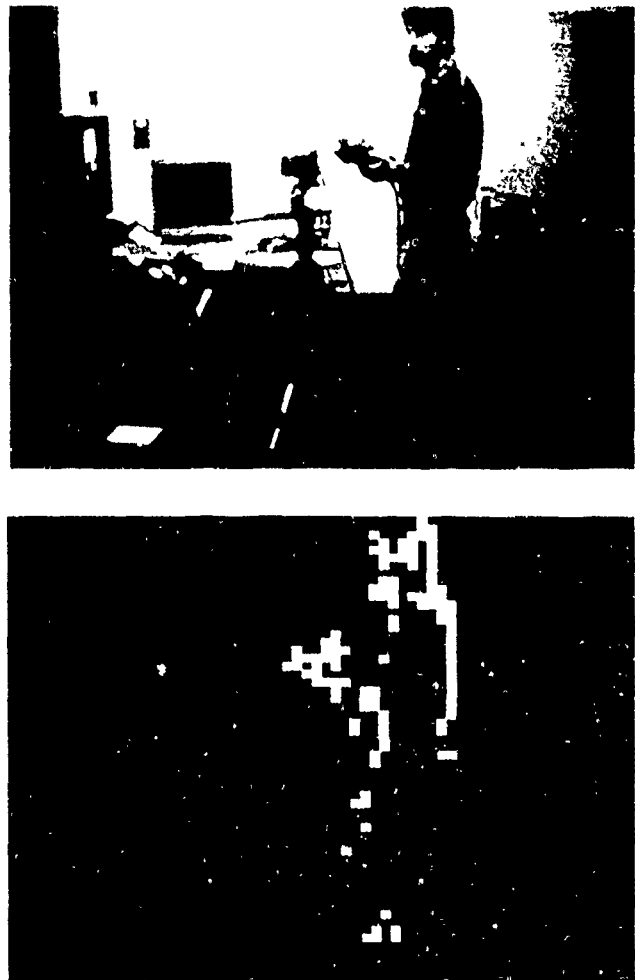


Figure 4: Detection of a walking figure in a moving scene by a movement detector based on constraint ray matching. The camera motion is a combination of rotation and translation whose net effect is to provide impart upward apparent motion (at velocities which depend on distance) to objects in the scene. The walking figure is detected as inconsistent with a rigid interpretation of the motion. The system operates in real time (10 frames/sec).

identifies those regions whose motion is inconsistent with a rigid interpretation of the world.

4. Detection of Animate Motion

4.1 Theoretical basis

The constraint ray filter described above depends on having some knowledge of observer motion. It is also possible to use knowledge about the motion of the object to be detected. In particular, we take advantage of the fact that, for an observer moving smoothly with respect to a rigid environment, the apparent motion of a world point projected on the image sphere is a relatively slowly changing function of time. Independently moving objects such as people, animals or rolling rocks, on the other hand, frequently *maneuver*, that is, they or their component parts follow trajectories for which the projected velocity changes rapidly compared to the apparent velocity change due to self motion. This suggests that high rates of change or temporal discontinuities in the the projected velocities of world points could provide a basis for distinguishing a wide variety of moving objects against an apparently moving background. Since the types of motion which would be detected by this method are characteristic of living creatures (though they are not the only source) we will use the term *animate motion* to refer to highly accelerated movement used in this context.

The intuitive argument presented above can be formalized as follows. Consider an observer translating with velocity (v_x, v_y, v_z) and rotating at $(\omega_x, \omega_y, \omega_z)$ with respect to the local Euclidean coordinate system established by the projection of world point p on the image sphere at time 0 (p projects to $(0,0)$ at $t=0$). The apparent acceleration of the projection of p in terms of this local image plane contains two terms: a coriolis term arising from the interaction of the apparent translation of the projection of p with ω_z , and a divergence term arising from the apparent expansion of the image due to translation in the Z direction. In component form the (angular) acceleration in the local coordinate system is

$$a_x = \omega_z(-\omega_x - \frac{v_y}{Z}) - 2 \frac{v_z v_x}{Z^2}$$

$$a_y = \omega_z(-\omega_y + \frac{v_x}{Z}) - 2 \frac{v_z v_y}{Z^2}$$

The first term in each expression is the coriolis effect; the second is the divergence.

We can use these expressions to determine the conditions under which the method is usable. Examining the acceleration equation, we observe that the accelerations due to self motion are on the order of the (angular) velocity of points in the image squared. The accelerations of independently moving objects, on the other hand are on the order of their (projected) angular velocity times the characteristic frequency of their movement. Thus, for example, if the components of the motion field due to self motion and independent motion are of comparable magnitude, the accelerations due to autonomous motion will stand out if

objects reverse themselves (180 degree phase shift) significantly faster than they traverse 180 degrees on the image sphere. This condition holds in a large variety of real-world situations.

The above analysis holds for a spherically projected image. In a planar image, there will be an additional acceleration induced by planar distortion at all points away from the image center. By twice differentiating the expression for planar projection we can show that the planar projection of a point moving away from the image center with apparent angular velocity v displays an apparent linear acceleration given (in one dimension) by

$$R\omega^2 \frac{\sin(\theta)}{\cos^3(\theta)}.$$

Where R is the distance from the center of projection to the image plane, and θ is the angular distance of the projection of p from the image center. Since $\sin(x)/\cos^3(x)$ is less than unity for $x < 34$ degrees, as long as the images are smaller than about 70 degrees square, this effect is smaller than those already mentioned.

4.2 Practical considerations

We next address the problem of identifying highly accelerated regions. Because of the effects of occlusion and depth discontinuities, simply differentiating the motion field with respect to time will not work. Conceptually, the problem can be solved by tracking the projections of world points from image to image, but actually doing this is often difficult as it requires solving a correspondence problem. Fortunately however, identifying regions where rapidly changing motion is present is simpler than obtaining quantitative values for the accelerations. The idea is to use the measured motion field at a point in the image to predict where associated world point might project in the next frame. Since the image motion due to non-maneuvering objects changes slowly, the candidate locations can be flagged to indicate that motion as a possible value. If the motion field were known precisely, then each point in the original image would flag a unique location and direction in the next frame. In general, however, since the field is inexact known, a multi-dimensional "footprint" of non-zero volume, whose exact shape depends on the nature of the available information, should be flagged. This makes even incomplete information, such as the gradient parallel component available from local differential measurements, usable. Carrying out this operation for each point in the original image produces a constraint map which lists possible values of the motion field for each point in the new image. Typically, since footprints from different antecedent points can overlap, a point in the map may contain more than one value. This constraint map can be compared to the computed field in the new image, and inconsistent points marked. These represent potential regions of high acceleration.

The above approach can yield a false negative if, due to the local complexity of the original motion field, so many different directions occur close together that their overlapping footprints obscure genuinely new values due to

changing motion. For most scenes, however, such regions constitute a small portion of the image if they occur at all, so this will generally not be a big problem. The approach can yield a false positive only at occluding boundaries when previously invisible points appear. Since they were not present in the original image to flag their future location, such points can produce spurious indications of changing motion. This problem can be greatly ameliorated by extending the footprint through and slightly to the counterflow side of its generating point. Thus an object which is partially visible and emerging from behind an occluding object will predict the appearance of similarly moving points at the boundary. The only case where this will break down is on the first appearance of such an object. Such events occur infrequently enough that they do not generally cause a problem and, in fact, represent situations which should be noticed, since a suddenly appearing object may very well be an independently moving one.

The animate motion method of has the advantage that it does not require any information about the observer motion, and is thus applicable for any smooth observer motion rather than just a subset. On the other hand, it can detect moving objects only when they maneuver. For animals, this is almost any time they move, since legs or wings must move back and forth to provide propulsion. Certain manmade targets such as ships and airplanes, on the other hand, may move at the same velocity for long periods. In this case, the technique would be inappropriate. The method would also be sensitive to jitter produced by small rotations of the observer, and thus requires some method of rotationally stabilizing the gaze. It is interesting to note in this connection, that animals which rely much on their eyes almost always possess some such system.

4.3 Real-time implementation and testing.

We have implemented a version of the animate motion detector described above. The first stage is similar to that utilized in our implementation of the constraint ray algorithm described in Section 3, with Datacube boards arranged to compute gradient parallel components of the motion field. This information is subsampled as before, and downloaded to the Sun, which computes the constraints at each pixel of the image using the footprint method described above. These constraints are encoded in an intrinsic image, which is used to filter the next image for motion which violates the temporal smoothness constraints. The algorithm runs in real time (about 10 hertz) robustly identifies animate (e.g. human) motion while the camera translates and rotates in a complicated 3-D environment. Figure 5 shows the detection of a moving hand from a moving camera. Unlike our implementation of the constraint ray filter, this method is not restricted to a limited set of observer motions, and seems to perform equally well under a very wide range of movements, the only criterion being that they not be too violent (in the sense quantified in section 4.1). The limitation of course, is that the system is insensitive to smoothly moving objects. Combining the systems could provide the best of both worlds, with the constraint ray algorithm operating providing detection of

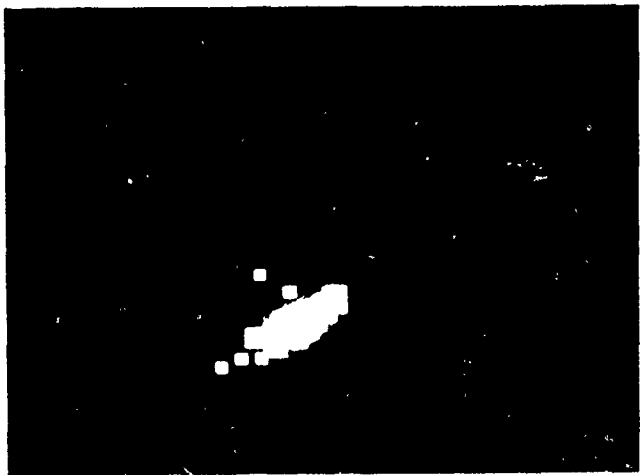
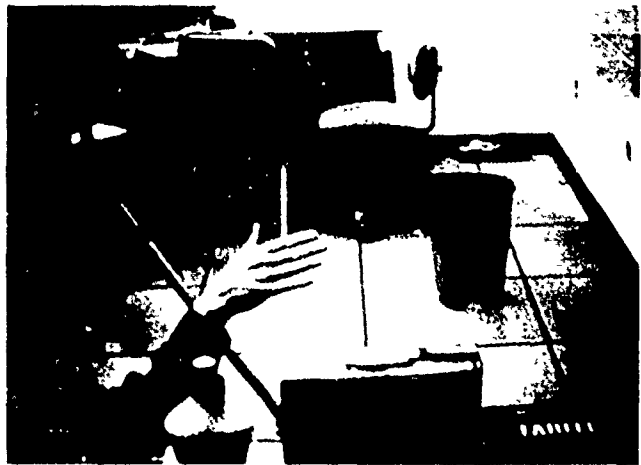


Figure 5: Detection of animate motion. The camera is translating to the left, which means that all the objects in the scene are apparently moving to the right with velocities that depend on their depth. The waving hand, however, can still be detected. The procedure runs in real time (10 frames/sec).

smooth independent movement when the observer motion is known.

5. Conclusions

We have described two methods for the detection of independently moving objects by a moving observer. The methods are robust in the sense that they are both resistant to error in the input, and can make use of motion information of low accuracy. This robustness results in large part from the use of matching and filtering techniques based on qualitative features of the motion field rather than numerical computations based on quantitative measurements. The methods are not infallible, in fact, as mentioned in section

3, there exist situations in which no method involving passive monocular observation can distinguish autonomous movement from apparent motion due to observer egomotion. However, it is possible to characterize precisely the circumstances under which the techniques are effective, and such analysis indicates that they have a broad useful range. Moreover, the domains are somewhat complementary. The first uses information about the motion of the observer, while the second make use of information about the motion of the object of interest. The techniques are primarily useful because they are extremely fast, and can thus serve as interest indicators to direct more sophisticated (and expensive) processing to critical areas. We envisage such detectors being used as the first of three steps in a general purpose motion recognition system. The second step involves stabilization of the area of interest through active visual processes such as fixation and tracking [Brow88b, Brow89, Coom89]. This places the motion of interest in a canonical form that facilitates the final recognition procedure. The third step, is the recognition of the region of interest via a more detailed analysis of its motion. We are currently engaged in developing such a motion recognition system.

References

- [Adiv85] - G. Adiv, Inherent ambiguities in recovering 3-D motion and structure from a noisy flow field, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1985, 70-77.
- [Anan85] - P. Anandan and R. Weiss, Introducing a smoothness constraint in a matching approach for the computation of optical flow fields, *Proc. Third Workshop on Computer Vision: Representation and Control*, 1985, 186-194.
- [Ande85] - C. H. Anderson, P. J. Burt, and G. S. van der Wal, Change detection and tracking using pyramid transform techniques. *Proc. SPIE Conference on Intelligent Robots and Computer Vision*, Boston MA, 1985, 300-305.
- [Barn80] - S. T. Barnard and W. B. Thompson, Disparity Analysis of Images, *IEEE Trans. PAMI* 2, 4, 1980, 330-340.
- [Bhan89] - B. Bhanu, P. Symosek, J. Ming, W. Burger, H. Nasr, and J. Kim, Qualitative motion detection and tracking. *Proc. DARPA Image Understanding Workshop*, 1989, 370-398.
- [Boll87] - R. C. Bolles Epipolar Plane Analysis: an Approach to Determining Structure from Motion, *Proc. International Joint Conference on Artificial Intelligence*, 1987, 7-15.
- [Brow88a] - C. M. Brown (Ed), with D.H. Ballard, T.G. Becker, R.F. Gans, N.G. Martin, T.J. Olson, R.D. Potter, R.D. Rimey, D.G. Tilley, and S.D. Whitehead, The Rochester robot, TR 257, Computer Science Dept., U. Rochester, August 1988.
- [Brow88b] - C. M. Brown and R.D. Rimey, Coordinates, conversions, and kinematics for the Rochester Robotics Lab, TR 259, Computer Science Dept., U. Rochester, August 1988.
- [Brow89] - C. M. Brown, Centralized and decentralized Kalman filter techniques for tracking, navigation and control, *Proc. DARPA Image Understanding Workshop*, May, 1989, 651-675.
- [Burt89] - P. J. Burt, J. R. Bergen, R. Hingorani, R. Kolczynski, W. A. Lee, A. Leung, J. Lubin, and H. Shvayster, Object tracking with a moving camera, *Proceedings of IEEE Workshop on Motion*, Irvine CA., 1989.
- [Coom89] - D. J. Coombs, Tracking objects with eye movements, *Proc. Topical Meeting on Image Understanding and Machine Vision*, Optical Society of America, 1989.
- [Heeg87] - D. Heeger, Optical flow from spatio-temporal filters, *Proc. 1st International Conference on Computer Vision*, 1987, 181-190.
- [Heeg88] - D. Heeger and G. Hager, Egomotion and the stabilized world, *International Conference on Computer Vision*, Tampa, 1988, 435-440.
- [Horn81] - B.K.P. Horn and B.G. Schunk, Determining optical flow, *Artificial Intelligence* 17, 1981, 185-204.
- [Litt88] - J. J. Little, H. H. Bulthoff, and T. Poggio, Parallel optical flow using local vote counting, *2nd International Conference on Computer Vision*, 1988, 454-459.
- [Long81] - H.C. Longuet-Higgins, A computer algorithm for reconstructing a scene from two projections, *Nature*, 293, 1981.
- [Moro79] - H. P. Morovec, Visual Mapping by a Robot Rover, *Proc. IJCAI* 1979, 598-600.
- [Nage33] - H. H. Nagel, Displacement vectors derived from second order intensity variations in image sequences, *Computer vision, Pattern Recognition, and Image processing*, 21, 1983, 85-117.
- [Nage86] - H. H. Nagel and W. Enkelmann, An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Trans. PAMI*, 85, Sept. 1986, 565-593.

[Nels88a] - R.C. Nelson and J. Aloimonos, Finding motion parameters from spherical flow fields (or the advantages of having eyes in the back of your head) *Biological Cybernetics*, 58, 1988, 261-273.

[Nels88b] - R. C. Nelson *Visual Navigation*, Ph.D. Thesis, University of Maryland, 1988, also University of Maryland Computer Science Department TR 2087.

[Nels89] - R.C. Nelson and J. Aloimonos, Using flow field divergence for obstacle avoidance in visual navigation, *IEEE transactions on PAMI*, 11, 10, Oct. 1989, 1102-1106.

[Praz81] - K. Prazdny, Determining the Instantaneous Direction of Motion from Optical Flow Generated by a Curvilinear Moving Observer. *Computer Vision Graphics and Image Processing*, 22 1981, 238-248.

[Schu84] - B. G. Schunck, Motion segmentation by constraint line clustering, *IEEE Workshop on Computer Vision: Representation and Control*, 1984, 58-62.

[Thom86] - W.B. Thompson and J. K. Kearney, Inexact vision, *Workshop on Motion, Representation, and Analysis*, May 1986, 15-22.

[Thom90] - W. B. Thompson and T. C. Pong, Detecting Moving Objects, *International Journal of Computer Vision* 4, 1, 1990 39-58.

[Tsai81] - R. Y. Tsai and T. S. Huang, Estimating 3-D motion parameters of a rigid planar patch I, *IEEE ASSP*, 30, 1981, 525-534.

[Tsai84] - R.Y. Tsai and T.S. Huang, Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces, *IEEE Trans. PAMI*, 6, 1984, 13-27.

[Ullm79] - S. Ullman, The interpretation of structure from motion, *Proceedings of the Royal Society of London*, B 203, 1979, 405-426.

[Uras88] - S. Uras, F. Girosi, and V. Torre, A computational approach to motion perception, *Biological Cybernetics*, 60, 1988, 79-87.

[Verr87] - A. Verri and T. Poggio, Against quantitative optical flow, *International Conference on Computer Vision*, June 1987, 171-180.

[Waxm87] - A. Waxman, Image Flow Theory: a Framework for 3-D Inference from Time Varying Imagery, *Advances in Computer Vision*, C. Brown (Editor), Lawrence Erlbaum Inc. 1987

Multiple Frame Analysis of Translation Dominant Motion

Yong Cheol Kim and Keith Price*
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, California 90089-0273

Abstract

An approach to multiple frame analysis for translation dominant motion is presented. It is generally known that 3 dimensional motion analysis algorithm is sensitive to noise. A long sequence of image frame generally provides redundant information. This redundancy can improve robustness to noise and thus soothe inconsistencies among the partial interpretations. Focus of expansion is computed from the region matches, refined by corner matches. Motion parameters and depth map for regions are computed from FOE. Regions with similar motion vectors are grouped together under the assumption that there are a multiple number of independently moving objects. In the motion analysis, higher priorities are given to data satisfying the assumption of translational motion, the result of which guides the analysis of the noisy regions in the same motion group.

1 Introduction

One of the prime research areas in computer vision is motion analysis using multiple images. The goal is to recover the relative motion between a viewer and the environment as well as the structure of the environment. Over the last decade, there have been many works in motion analysis, most of which are based either on the establishment of feature correspondences between frames [Broida and Chelappa, 1986; Shariat and Price, 1990; Tsai and Huang, 1984], or the estimation of the intensity velocity of points [Adiv, 1985]. Both of them are computationally expensive and sensitive to noise. With noise added to the optical flow or feature correspondence, it is difficult to correctly estimate the motion parameters.

There have also been approaches in which some constraints are imposed on the motion and relatively simple analysis is done on the restricted motion in order to speed up the computation and improve robustness to noise. It is reported that a simplified analysis may result

in a considerable error in the estimation of the motion even with a slight deviation of the assumption of motion [Snyder, 1989].

It is often found that an algorithm that works in the absence of noise has difficulty in accurately estimating motion at a realistic level of noise [Tsai and Huang, 1984]. Interference of noise are unavoidable in dealing with real image data. At the front end of image acquisition, the physical limit of sensory systems introduces measurement noise to data, e.g. errors introduced by the digitization process. Variations in the extracted feature set from frame to frame may decrease the reliability of the data. The most common source of noise is in feature matching. Establishment of correspondence is subject to some matching errors since it is essentially identical to an incomplete graph matching. The possibility of incorrect matching increases for multiple frames leading to breaks in the sequences of matches. A strict screening on correspondence would decrease not only erroneous data but also useful data.

A more reliable result can be expected when a global approach is used, where motion analysis is done as a part of an integrated process rather than as a single isolated process. An integrated process means the combination of all phases of motion interpretation, including segmentation of an image, feature extraction, feature matching, motion parameter estimation, and three-dimensional structure estimation.

A global approach in the motion interpretation of a sequence of images should control two tasks. One is the control of the interactions among the phases of the integrated process. The global module and its subsidiary modules exchange information in both directions. For example, segmentation and motion estimation are so closely related that both of them can be improved with feedback from one to the other.

The other task is the coordination of partial interpretations. A long sequence of image frames generally provides redundant information for motion analysis. The motion is consistent throughout the observation. The necessary information can be extracted from a part of the whole sequence. If an object is moving, a small subset of them may supply the information. However, there can be many partial interpretations in a long sequence, which should be consistent with each other.

*This research was supported by the Defense Advanced Research Projects Agency under contracts F33615-87-C-1436 and F49620-89-C-0126, monitored by the Wright-Patterson Air Force Base and the Air Force Office of Scientific Research, respectively.

However, separate partial interpretations of noisy data are often inconsistent and mutually contradictory. With an appropriate management of redundant data, this inconsistency may be remedied, or at least a confidence factor for each interpretation may be computed.

Three basic ideas are incorporated into our integrated motion analysis system:

- Get a rough estimation using data less sensitive to noise.
- Verify and correct rough estimates.
- Make use of all information available.

Our integrated motion analysis system is being used to generate a rough description of three-dimensional structure of the environment, using matches for region-based features, with refinements from matching corner features. All of the processing is applied over multiple (at least 5-10) frames. The current effort is on computing a depth estimate when the dominating motion is translation along the axis of the camera.

An estimate of the FOE (focus of expansion), which is computed from the region or corner matches, guides the computation of the direction of motion. The estimates of the motion parameters and depth map on each region are monitored globally. Priority is given to regions with clean data. Regions with inconsistent parameters go through a guided analysis where the result of the more reliable ones give some constraints to the motion parameters.

A brief outline of data analysis is given in the next section. In subsequent sections, feature matching, estimation of the FOE and motion vector, grouping and depth estimation of regions are described. The results of experiments on real image sequences are given in section 6.

2 Sequence of Data Analysis

The block diagram of the integrated motion analysis system is shown in figure 1.

Each image is segmented into a set of uniform regions using a recursive region splitting technique [Ohlander *et al.*, 1978], where the segmentation for the previous image frame guides the segmentation for the current image. This guidance helps to insure that the segmentations are somewhat consistent from frame to frame.

Feature matching is performed in 2 steps. First, region correspondences over multiple frames are computed. Then correspondences are established for corner points extracted from the contours of the matched regions. Each region receives a reliability factor that measures the linearity of the corner matching associated with the region, i.e. this measures the similarity to translational motion. Matching data with low linearity are considered to be erroneous. Higher priorities are given to data satisfying the assumption of translational motion. Motion analysis is first performed on regions with high reliability factors, which guides the analysis of the others. More details of this work are described in section 3.

The next step is to compute the FOE for those regions with a high reliability. For an object under pure

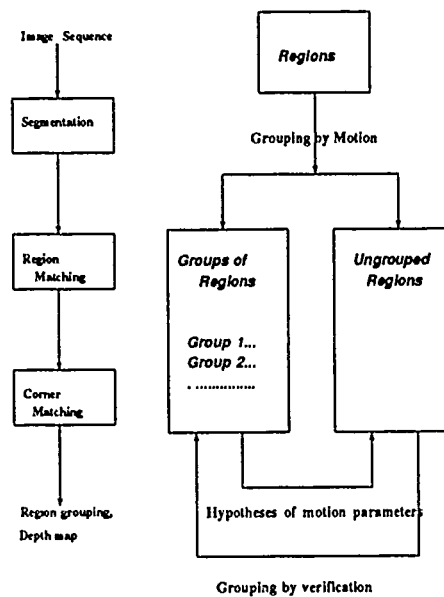


Figure 1: Sequence of data analysis

translation, both the direction of the translation vector and the relative depth of the object are computed from the position of FOE in the image plane. The position of FOE for each region is verified using the computed three-dimensional structure. A constant incremental depth from frame to frame is the criterion in the verification step. Regions with validated FOE's are grouped according to the direction of their translation vectors. This assumes that there may be multiple independently moving groups of objects. The motion groups act as a guide for similar analysis of the noisy regions.

The analysis for the regions with low reliability factor is based on the hypothesis and test paradigm. The hypotheses are the motion parameters for each motion group and the three-dimensional structure for the noisy region computed from these parameters. The region is merged into the motion group that produces the most constant incremental depths from frame to frame.

Finally, the depth is computed over the surface of the matched regions in the motion groups, with the surfaces approximated as planar patches. This surface reconstruction step is not the thrust of the effort and is one of many different techniques for such analysis.

3 Feature Matching

Feature matching is done in an hierarchical manner to reduce computation time and increase stability. First, region correspondences are established for adjacent frames in the sequence. Then point correspondences are obtained for the corner points extracted from the bound-

aries of the matched regions. A single relaxation-based matching routine [Faugeras and Price, 1981] is used for matching both sets of features (regions and corners) by changing the feature set used by the matching routine. Corner matching for a region is guided by the motion computed for the center of mass of the matched regions and by the fact that corresponding corners must appear on corresponding regions. This produces stable feature correspondences over multiple frames of a sequence.

For convenience, we define an RMS (region matching sequence) as a sequence of matched regions over multiple frames and a CMS (corner matching sequence) as a sequence of matched corners based on an RMS. In figure 3 and figure 6, CMS's over an RMS are shown for each frame sequence used in the experiment. The region boundaries are shown with dots indicating the corners in the CMS. The approximate motion of a region can be computed from an RMS, and the fine structure can be computed from the CMS's of the region.

3.1 Reliability of Matching Sequences

There are many factors to decrease the reliability of a matching sequence. If an RMS contains errors, then the CMS's based on it will be erroneous, too. There are also fluctuations in the detected location of the corners from frame to frame as can be seen in figure 3. Even if an RMS is correct, the CMS's are subject to matching errors. Because of these problems it is necessary to measure the reliability of a matching sequence and to depend on the most reliable measurements.

The reliability factor of a CMS is related to how many image frames are included and how many CMS's occur in an RMS. Without occlusion, a CMS for a prominent corner should contain as many matches as the associated RMS. A missing match for a corner at some frame breaks a long corner matching sequence into two CMS's. So, a higher reliability is given to a longer CMS. The number of CMS's for an RMS depends on the size and shape of the region associated with the RMS. Usually several reliable CMS's are available for a region. But if the RMS is unstable (or has errors) or the region does not have prominent corners, few reliable CMS's are available. So, a higher reliability is given to an RMS that contains more CMS's.

When a point undergoes a pure translation in 3 dimensional space, its depth change is constant and its image lies on a straight line in the image plane. This leads us to choose the linearity of matched sequences as a reliability measure. The 2-linearity is a measure of the linearity of a matching sequence in the image plane and is used to choose more reliable ones. For a CMS, 2-linearity is defined as the product of its length and the fitness when approximated by a line segment. 3-linearity measures the uniformity of incremental depth from frame to frame and is used to verify estimated parameters. The 3-linearity of a CMS is then defined as the standard deviation of depth difference normalized by the mean value of the difference.

$$2linCMS = length(CMS) \frac{D}{M}$$

where D and M respectively represent the average dis-

placement of a point in a CMS and the mean square error when the CMS is approximated by a straight line.

$$3linCMS = \frac{M}{D}$$

where M and D represent the average and standard deviation of incremental depth of the corner.

The 2-linearity and 3-linearity of an RMS is defined as the weighted sum of the measures of its best three CMS's. Hence preference is given to an RMS with more than three CMS's.

$$2linRMS = \sum_{i=1}^3 2linCMS(i)w(i)$$

$$3linRMS = \sum_{i=1}^3 3linCMS(i)w(i)$$

where $w(1) = 27$ $w(2) = 4$ $w(3) = 1$.

4 Estimation of the FOE and Translation Vector

It is well known that the two-dimensional (image plane) flow vectors of an object under pure translation all pass through a single point, the FOE. The same also holds for sets of matching feature points. Ideally, all the CMS's intersect, when extended, at the FOE. As stated in section 3, CMS's may have erroneous components, thus spreading the intersection point for the CMS's over a wide range.

Other work on the selection of an estimate of the FOE from a set of possible values includes using a modified Hough transformation [Manmantha *et al.*, 1989] or by searching [Bhanu *et al.*, 1989]. We chose to use a weighted average in LMSE sense. In the average, the intersection points for a set of unreliable CMS's were discarded to minimize the spuriousness of erroneous matches. The weighting factor is given by:

$$weighting factor = CA \times CB \times \sin(\theta)$$

where CA and CB are the 2-linearities of the CMS's for the intersection point and θ is the angle between the CMS's. Intersections with a weighting factor below a given threshold are not included in the average, since they may have a large deviation from the exact FOE position that influences the result even though the weighting is small.

When the position of the FOE is known, the direction of the motion vector of a translating object is easily computed. Let the normalized motion vector be $\vec{T} = (T_x \ T_y \ 1)$ and the FOE lies on $(V_x \ V_y)$ in the image plane. Then

$$T_x = -\frac{V_x - C_x}{F}$$

$$T_y = -\frac{V_y - C_y}{F}$$

where (C_x, C_y) is the center of the image plane and F is the focal length.

The environmental depth of a point can be obtained by a simple function from [Snyder, 1989],

$$\frac{Z_{i+1}}{T} = \frac{r_i}{r_{i+1} - r_i}$$

where Z_i is the environmental depth of a point at frame i and r_i is the distance from the FOE of an image point at frame i . T is the unit advance (scaling factor) of the object and can be set to any nonzero value if unavailable. When a point correspondence (e.g. CMS) of length N is available, the depth at each frame is obtained by

$$Z_k = \frac{\sum_{i=0}^{N-2} r_i r_{i+1}}{r_k(r_{N-1} - r_0)}$$

Thus the depth is inversely proportional to the distance of a point from the FOE position.

5 Grouping and Depth Estimation

Since the interpretation of sequences with noisy data can be guided by that of clean ones, analysis is performed first on reliable regions and then on the other regions. First a set of regions with a high value for the 2-linearity of the RMS is selected. The FOE is computed for these regions and verified by the computed depth of the region. Using the assumption of uniform translation, the incremental depth from frame to frame does not change. The FOE of a region is considered correct if the variation in incremental depth associated with that FOE is small. Those regions with verified FOEs are classified into motion groups by similarity of the direction of their translation vectors. The grouping of the other (unreliable) regions are guided by this motion group. The block diagram of region grouping is shown in figure 1.

This Self Grouping step applies for all reliable regions and generates a translation vector that represents the motion for all regions within the group. Any number of independently moving object groups are allowed in the image sequence.

Supported Grouping uses the motion parameters of the motion groups formed from the self grouping procedure as a guide for valid motion groups. If at least one reliable correspondence from each motion group is available, then it is highly probable that any moving object is in coherent motion with one of the motion groups. For each of the the ungrouped regions, the FOE position computed for each motion group is used to compute the depth for the region and to choose the motion group which produces the least variation in incremental depth. If the variation is within a given threshold, the region is merged into that motion group.

After self grouping and supported grouping, some regions may still remain unclassified. The most common ones include: a region with a small disparity that cannot be analyzed correctly, and regions where accumulated noise from all the phases of image analysis block the correct interpretation.

6 Result

The integrated system has been tested for 2 image sequences. One is a decimated (each tenth frame) sequence

frame 120-frame long hallway sequence from SRI. The length of the analyzed sequence is 12 and the image size is 256 by 240. The other one is a car sequence consisting of 6 frames during which a car moves from the right side of a 760 by 512 image to the left side.

Figure 2 through figure 4 shows the result for the hallway sequence. The image of the 6-th frame of the sequence (51st from the original) and its region segmentation are shown in figure 2. As shown, the average number of regions from segmentation is around 10. CMS's superimposed on an RMS is shown in figure 3 with their estimated FOEs as large dark circles. The estimated depths for the best three corners are shown in table 1. Using a planar approximation of the surface of a region, depth interpolation was performed for the surface of the regions from these three corners. The result of interpolation is shown in figure 4.

The results for the car sequence is shown in figure 5 through figure 7. The image of the 4-th frame and its region-based segmentation are shown in figure 5. CMS's superimposed on an RMS is shown in figure 6 with their estimated FOEs. As shown in fig, the average number of regions from segmentation is around 7. A 3 dimensional plotting of depth of the surfaces of regions is shown in figure 7.

7 Conclusions

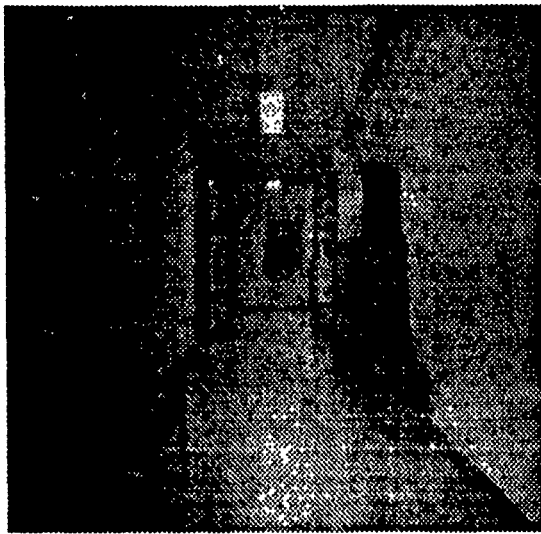
In this paper, we described a method of multiple frame analysis for translation dominant motion and showed the result for 2 real image sequences. It has been shown that redundant information in a long sequence of images can be used to improve robustness to noise and coordinate inconsistencies among partial interpretations. We have also shown that a proper handling of noisy correspondence data can result in fairly good estimation of FOE and motion vector by giving appropriate reliability to each feature match.

Motion analysis heavily relies on grouping of regions based on hypothesis and test paradigm. Regions with clean data are grouped in accordance with their motion vectors. The motion parameters of motion groups are used as hypotheses in the analysis of the noisy regions. We could get a depth map of regions, which fairly well represents the relative depth of the environments.

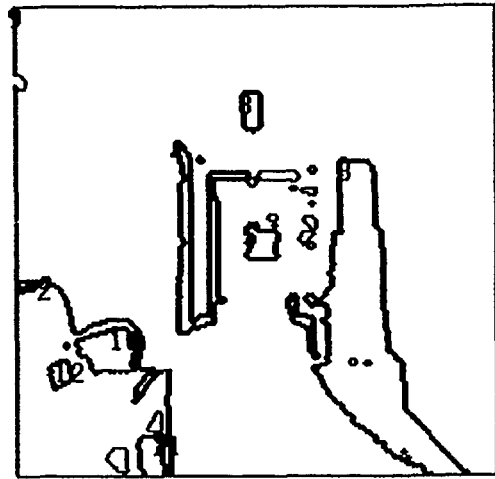
This system is working well at a reasonable level of noise, including matching error and localization error in feature extraction. However, there still remain some unidentified regions, which are either very noisy or too far from the camera.

References

- [Adiv, 1985] Gilad Adiv. Determining 3 dimensional motion and structure from optical flow generated from several moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(4), july 1985.
- [Bhanu et al., 1989] Bir Bhanu, Peter Symosek, John Ming, Wilhelm Burger, Hatem Nasr, and John Kim. Qualitative target motion detection and tracking. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, California, 1989.



(a) Image from sequence



(b) Region segmentation for image

Figure 2: 6-th frame of hallway sequence

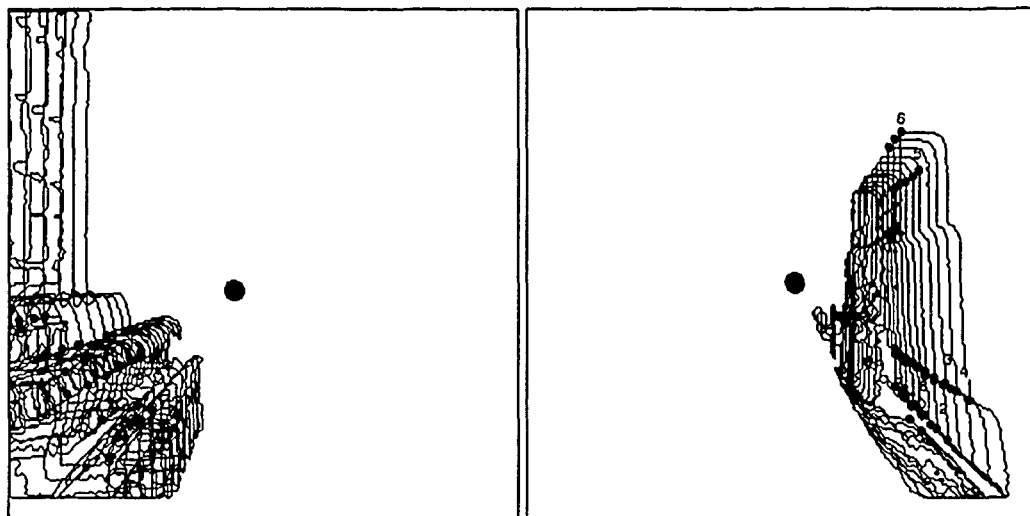
- [Broida and Chelappa, 1986] T. Broida and R. Chelappa. Estimation of object motion parameters from noisy images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, January 1986.
- [Faugeras and Price, 1981] O.D. Faugeras and K. Price. Symbolic description of aerial images using stochastic labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(6):633-642, November 1981.
- [Manmantha *et al.*, 1989] R. Manmantha, R. Dutta, Edward Riseman, and M. Snyder. Issues in extracting motion parameters and depth from approximate translational motion. In *Workshops on visual motion*, Irvine, California, 1989.
- [Ohlander *et al.*, 1978] R. Ohlander, K. Price, and R. Reddy. Picture segmentation by a recursive region splitting method. *Computer Graphics and Image Processing*, 8:313-333, 1978.
- [Shariat and Price, 1990] H. Shariat and K. Price. Motion estimation with more than two frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):417-434, 1990.
- [Snyder, 1989] M. A. Snyder. The precision of 3d parameters in correspondence-base technique: The case of uniform translation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5), may 1989.
- [Tsai and Huang, 1984] R.Y. Tsai and T.S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 13-26, January 1984. Vol. 6, No. 1.

frame-num	2	3	4	5	6	7	8	9	10	11
reg-2								09.89	08.88	07.89
					13.47	12.42	11.42	10.37	09.38	08.42
							10.98	09.89	08.95	07.95
reg-3								13.76	12.74	11.76
			23.89	22.92	21.97	20.93	19.83			
	22.80	21.84	20.70	19.67	18.74	17.89	16.84	15.85	14.80	13.77
reg-4	32.26	32.40	30.38	29.72	28.12	27.22	26.37	25.11		
							19.77	31.26	18.54	
reg-8								74.97	73.95	72.97
	32.77	31.85	30.99	29.79	28.67	28.30	26.96	25.75	24.64	23.85
	36.36	35.36	33.93	33.49	32.61					
reg-10		13.05	12.05	11.05						
								09.75	08.79	07.75
							09.96	08.96	07.95	07.07
reg-12							44.08	43.30	42.36	41.36

Table 1: Depth of the best 3 corners from each region of Hallway sequence

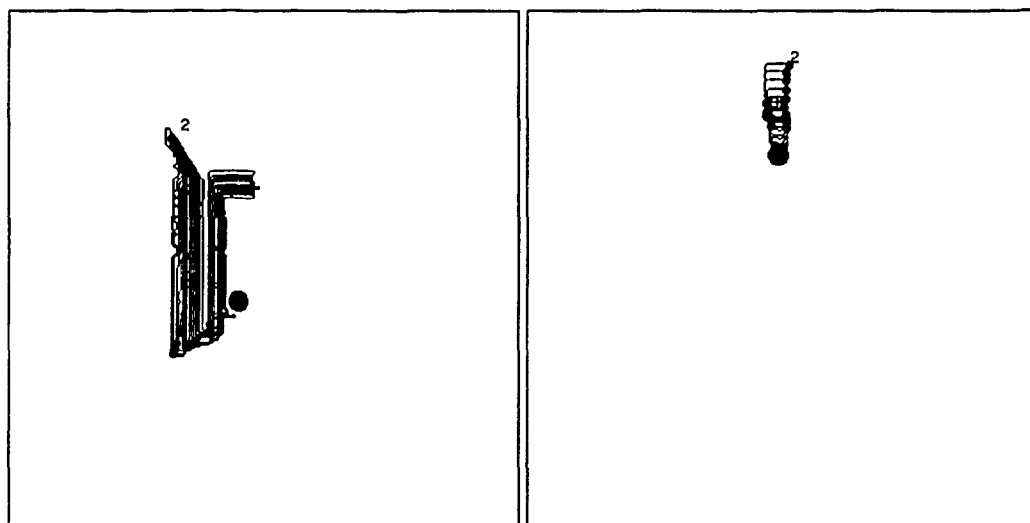
frame-num	0	1	2	3	4	5
reg-3				07.99	07.01	06.00
		10.44	09.40	08.38	07.40	06.41
		09.80	08.78	07.75	06.78	
reg-5	10.75	09.72	08.70	07.67	06.69	05.71
	10.89	09.83	08.81	07.81	06.81	05.84
	10.77	09.71	08.71	07.68	06.69	05.72
reg-6		10.05	09.04	08.00	07.00	06.03
		10.10	09.06	08.06	07.03	06.07
		11.01	09.97	08.81	07.99	06.97
reg-7		10.88	09.89	08.88	07.88	06.88
		10.85	09.86	08.81	07.84	06.84
		11.02	10.05	08.99	08.00	07.01

Table 2: Depth of the best 3 corners from each region of Car sequence



(a) Region 2

(b) Region 3



(c) Region 4

(d) Region 8

Figure 3: FOE from each region of hallway sequence

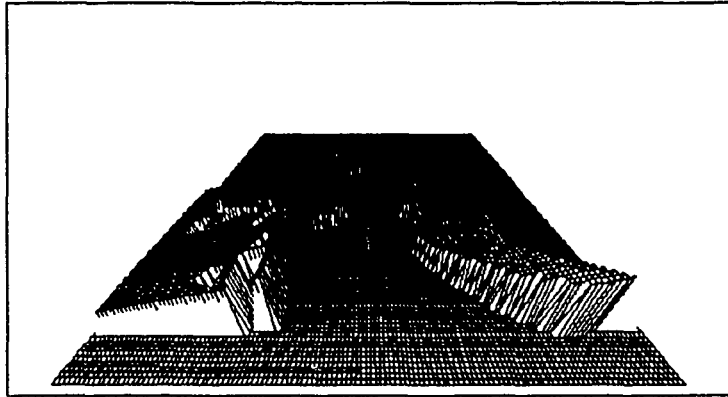
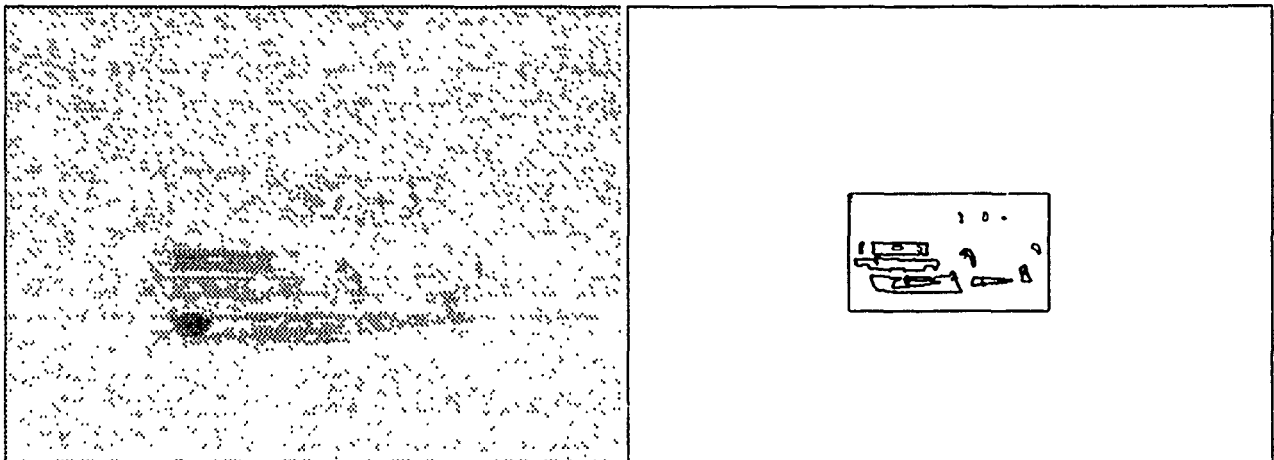


Figure 4: 3 dimensional depth of 6-th frame of hallway sequence



(a) Image

(b) Segmentation

Figure 5: 4-th frame of car sequence

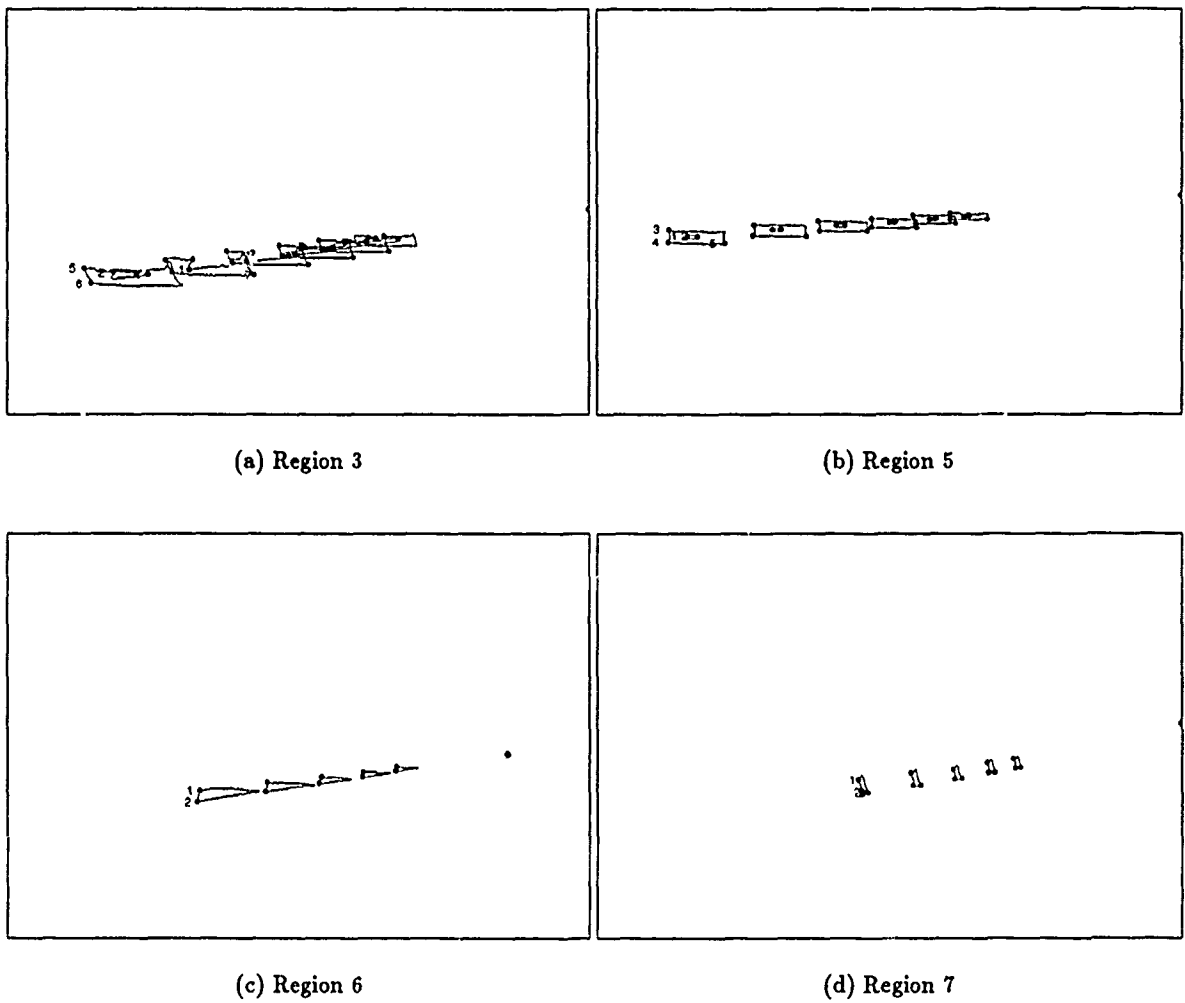


Figure 6: FOE from each region of car sequence

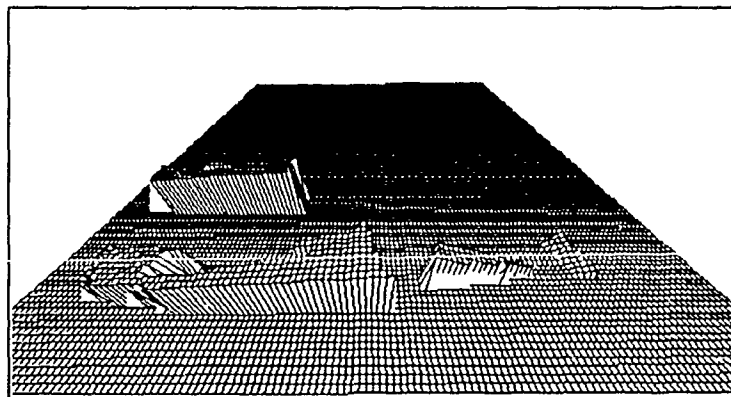


Figure 7: 3 dimensional depth of 4-th frame of car sequence

DETECTING MOVING OBJECTS FROM A MOVING PLATFORM

J. Frazier and R. Nevatia

Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, California 90089-0273

Abstract

We present a system that successfully detects edge segments of moving objects viewed by a translating camera. The system uses two key concepts. First, a Complex Logarithmic Mapping (CLM) is used to simplify detection of the desired movement. This converts the problem from one of detecting a complex motion along both the X and Y axes, to one of detecting vertical motion along an angular axis. Second, a mechanism to detect "vertically-moving edges" has been developed to detect the requisite vertical motion within the CLM image. Unlike other techniques, the method presented here does not require densely-sampled imagery. The system also provides insight into uses and organization of moving-edge detectors found in biological vision systems.

1 INTRODUCTION

Detecting moving objects from a moving platform has several important uses, including military and automotive applications. It is a difficult problem, however, because observer motion causes stationary objects to appear to move in the image. Thus, to detect moving objects, we must separate genuine motion from the apparent motion of the stationary environment.

We present preliminary results of a promising technique to solve this problem. The technique assumes that non-translational components of the observer's motion are very small.

To simplify detection, a Complex Logarithmic Mapping (CLM) is first performed. This converts the problem from one of detecting a complex object motion along both the X and Y axes, to one of detecting vertical motion along an angular axis. The CLM requires the location, projected on the image plane, of the distant point in space toward which the observer is moving. We assume this location (called the focus of expansion, or FOE) either is constant, or changes slowly.

A mechanism to detect "vertically-moving edges" has been developed to detect the requisite vertical motion within the CLM image. This moving-edge detector senses any vertical motion between frames that is greater than a specified distance threshold. Thus, the system can detect objects that move distances of many pixels per frame, and therefore does not require densely-sampled imagery.

The distance threshold is currently the only parameter (given the FOE) required by the system, and the system performs properly over a wide range of threshold settings. Preliminary

results indicate that, once the threshold is set high enough to eliminate false alarms, it can be increased by a *factor of five* and still correctly detect moving objects.

The moving-edge detector requires no object matching or recognition, and can thus detect moving objects that are partially occluded or camouflaged.

We show results of processing both synthetic and real image sequences. The synthetic sequence is used to illustrate the approach. The real sequence consists of imagery, containing a moving truck, taken from an observation vehicle that was following a road. The system correctly detects (with no false alarms) portions of the truck as it passes the observer. The system works even though (1) the truck moves a large distance between each image at the beginning of the sequence, (2) the size (in the image) of the truck varies considerably over time, and (3) the camera slowly tilts throughout the sequence.

2 PREVIOUS WORK

Several researchers have created systems to detect moving objects, in the presence of a moving background, from a monocular image sequence. [Zhu, 1988] and [Lee and Lin, 1988] apply a multiresolution approach. [Blostein and Huang, 1988] perform a tree search to match trajectories within a scene.

[Thompson and Pong, 1990] have implemented systems that track interest regions, and separate systems that utilize a constraint based on both optic flow and knowledge of scene depth.

[Heeger and Hager, 1988] utilize camera motion parameters (for translation and rotation) to find optical flow vectors (in the rectangular reference frame) that are inconsistent with the assumption of a stationary environment. They show results on synthetic imagery.

[Bhanu and Burger, 1988] apply a technique to qualitatively determine the focus of expansion. They then detect moving objects by assuming that they have a higher rate of translation in proportion to their size than arbitrary background regions.

[Jain, 1984] assumes a translating observer, and performs a Complex Logarithmic Mapping to simplify detection of moving objects. A series of accumulative difference images are computed, object segmentation is performed based on motion in CLM space, and CLM object trajectories are estimated. Moving objects are then detected based on their trajectories.

The technique presented here also uses the CLM, but with a simplified method to detect moving objects in CLM space that requires only two frames for analysis. The method used to detect the moving objects does not rely on traditional optical flow computations, which require densely-sampled imagery. Nor does the technique require segmentation of objects within the scene.

3 FOCUS OF EXPANSION

When an observer moves in a straight line, toward a distant point in space, stationary objects in the environment appear to move along paths radiating from that point. The point from which the paths radiate is called the *focus of expansion*, or FOE (see figure 1). We assume that non-translational components of the observer's motion are very small, which implies that the FOE and camera orientation are relatively stable between two successive images. In other words, the observer must not sharply turn or tilt between images.

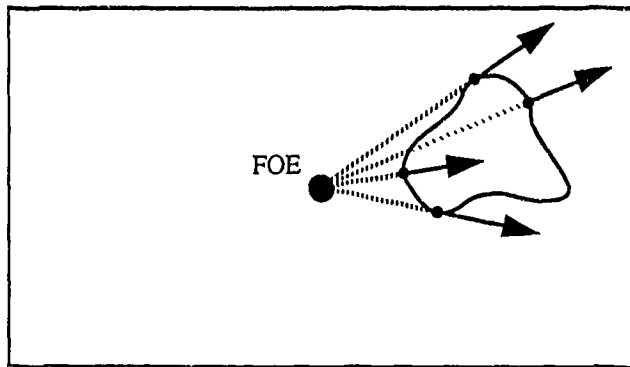


Figure 1: The paths of stationary objects radiate from the focus of expansion.

In the current system, the FOE is measured manually. Many systems for automatically computing the FOE have been developed in previous work, e.g. [Burger and Bhanu, 1989]. We are developing software to automatically compute the FOE as described in the Future Work section.

4 COORDINATE TRANSFORMATION

To simplify motion detection, we first transform the image to a different coordinate system. We could convert to polar coordinates, using the FOE as the origin:

$$r = \sqrt{(x - x_{FOE})^2 + (y - y_{FOE})^2}$$

$$\theta = \text{angle}(x - x_{FOE}, y - y_{FOE})$$

where r is the radial distance from the FOE to the rectangular image coordinate (x, y) , and θ is the angle (from 0 to 2π radians) subtended by the rectangular image coordinate (x, y) , the FOE, and the rectangular image coordinate $(1, 0)$.

That simplifies the problem because, in a polar system, stationary objects move through the image only in the direction of increasing radius (given the assumption of a translating observer), and their angular coordinate remains constant. Thus, any object that moves in the angular direction must be a moving object.

We use, however, a variation of the polar coordinate transformation called a Complex Logarithmic Mapping, or CLM [Jain and O'Brien, 1984]. This mapping differs from the conversion to polar coordinates only in that we use the logarithm of the radius:

$$r_{CLM} = \log(r)$$

$$\theta_{CLM} = \theta$$

where the CLM subscript indicates the Complex Logarithmic Mapping (versus polar) versions of the radius and angle variables.

The CLM provides scale, rotation, and projection invariances under certain conditions as described in [Jain and O'Brien, 1984]. It is also believed to approximate the retinotopic mapping in the human visual system [Schwartz, 1980]. Although these properties are not specifically required by the motion detection process, they are potentially useful for extensions to this work.

Figure 2 graphically illustrates the CLM. In the standard rectangular coordinate system (figure 2(a)), the white lines become brighter with increasing radius, and the black lines become darker with increasing angle. In CLM coordinates (figure 2(b)), we see the white lines of increasing radius, and the black lines of increasing angle. Notice the scalloping that occurs on the right side of the CLM image. The sharp points of the scallops correspond to the corners of the rectangular image, and the curves between the points correspond to the boundaries of the rectangular image.

Figures 3(a) and (c) show two synthetic images from a sequence depicting a road, horizon, billboard, and airplane. As the observer travels along the road (figure 3(a) to figure 3(c)), the billboard travels along a path radiating from the FOE, in this case the point at which the road meets the horizon. Unlike the billboard, the plane is not a stationary object, and its path does not radiate from the FOE.

Figures 3(b) and (d) show the corresponding images in the CLM coordinate system. The images show that the billboard, being stationary with respect to the environment, moves only in the direction of increasing radius (i.e., horizontally and to the right). This is in contrast to its motion in the rectangular reference frame, in which the billboard moves in both the horizontal and vertical directions.

The plane, on the other hand, is moving with respect to the environment. Its path does not radiate from the FOE, and therefore it exhibits a component of motion along the vertical (or angular) axis of the CLM image. Thus, to detect genuinely moving objects from a moving platform, we must detect vertical motion in the CLM coordinate system.

5 VERTICAL MOTION DETECTION

The technique described here to detect vertical motion is designed to detect vertical movement of horizontal (or near-horizontal) edges. Detecting vertical movement of non-horizontal edges is difficult because of the aperture problem (i.e., determining the component of an edge's motion parallel to the edge requires knowledge of the motion of an endpoint of the edge, which can be difficult to obtain). Several techniques have been developed to detect moving edges (e.g. [Kahn, 1988]). Many, however, assume densely-sampled im-

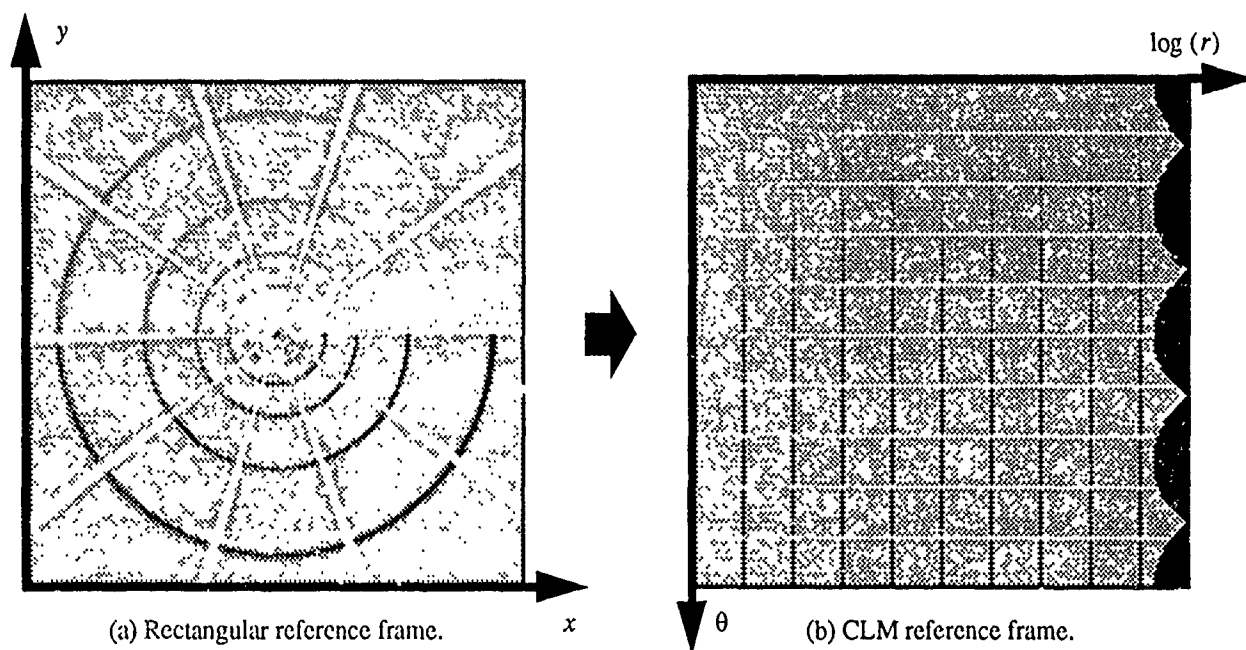


Figure 2: Image of radial lines and concentric circles transformed from rectangular to CLM reference frames.

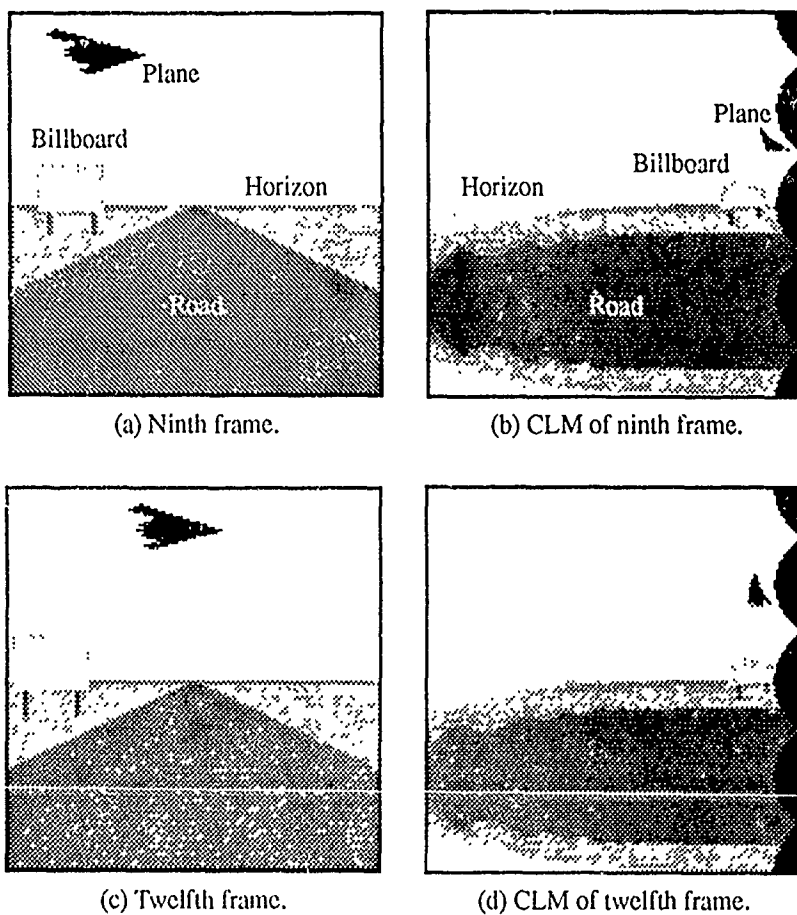


Figure 3. Two synthetic images, and corresponding CLM images, of a moving airplane and stationary billboard as viewed by a moving observer traveling along a road.

agery (e.g. that objects move less than one pixel per frame), whereas the approach described here does not.

The basic idea of our approach is as follows. If a horizontal edge in an image moves horizontally (figure 4(a)), then the overlap between the edge from one image to the next is very large. On the other hand, if a horizontal edge moves vertically (figure 4(b)), there is very little overlap, if any. The following technique first enhances horizontal edge components of the CLM image, then uses the idea of overlap to detect any vertical motion of those edges.

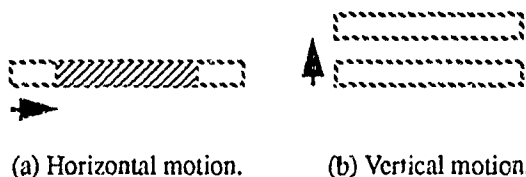


Figure 4: The overlap of a moving horizontal edge from one image to the next depends on the edge's direction of motion.

First, we convolve the CLM image with the horizontal edge mask of the Sobel operator. Figure 5(a) shows the effect of performing this convolution on the image of figure 3(b). The result is an estimate of the partial derivative of the CLM image, along the vertical direction.

Next, each pixel value of the partial derivative image is squared. This produces large positive values along horizontal edge components of the image. It forms a *reference image* of all horizontal edge components.

Then, we multiply the partial derivative of the current CLM image by that of the previous CLM image. This produces an image similar to the above reference image. The difference is that the result has large positive pixel values only along horizontal edge components that *overlap* from one image to the next. Edges that move vertically produce little overlap, and are eliminated.

This product is then subtracted from the reference image, producing a map (shown in figure 5(b)) of all horizontal edge components that have moved vertically since the previous frame. Actually, this map does contain small pieces of horizontally-moving edges that did not completely cancel out. In practice, however, these small pieces are very weak, and are filtered out by a thresholding process. The result, after thresholding, is shown in figure 5(c).

The threshold was set manually for the results presented here. We expect that it can be computed from image statistics (namely, from the correlation function and histogram of the horizontal Sobel image), and from the desired distance (in pixels) of vertical movement above which motion is to be detected. Initial results indicate that the system works properly over a broad range of threshold settings.

After thresholding is completed, the detected motion is transformed back to the rectangular reference frame (figure 5(d)), and then overlaid upon the original image. Figure 5(e) shows a dimmed version of the original image, with the bright white areas representing the detected motion. Motion was de-

tected around edges of the airplane in each of the 20 frames of the synthetic sequence, with no false alarms.

6 RESULTS ON REAL IMAGERY

Figure 6 shows images 4, 5, 8, and 15 of a real road sequence containing a truck passing a moving observer. Figure 7 shows the resulting detected motion overlaid upon the original images.

Notice that the truck moves a very large distance between the consecutive images of figures 6(a) and (b), illustrating that the imagery is not densely sampled. Also note the large change in the truck's size from figures 6(a) to (d). When viewed in rapid succession, the images also reveal a gradual rotation, or tilt, of the camera. Despite these circumstances, the system successfully detects portions of the moving truck (with no false alarms) in each of the 15 frames of the sequence. The FOE was determined manually (for one image in the sequence), and the same image coordinate was used for the FOE over the entire sequence.

Note that the system currently detects *portions* of moving objects, and that we plan to extend it to detect *complete* moving objects.

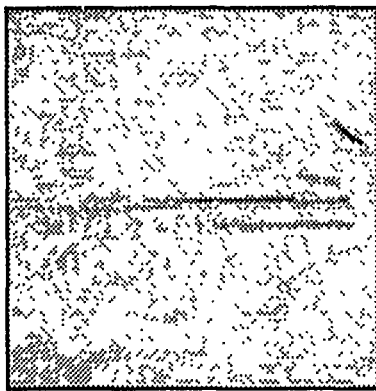
7 ADVANTAGES AND LIMITATIONS

The advantages of the system are:

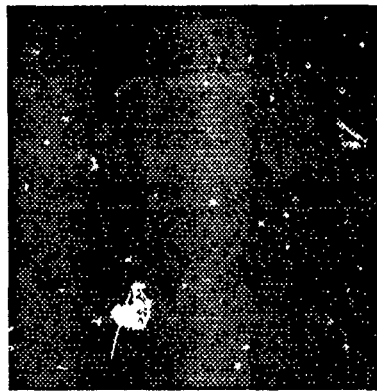
- It does not require densely-sampled imagery, meaning that objects may move distances of many pixels per frame without detrimental effects.
- It currently requires (assuming the FOE is given) only one parameter, the threshold used for vertical movement detection. Note that this version of the system does not include automatic computation of the threshold, and that the threshold was set manually for the results presented here. One threshold value was used for the entire plane sequence, and another threshold was used for the entire truck and road sequence. Preliminary results indicate that, for a given pair of images, once the threshold is set high enough to eliminate false alarms, it can be increased by a *factor of five* and still detect the moving object.
- The system requires only two images for analysis, which reduces sensitivity to gradual orientation changes.
- It is computationally efficient and highly suited to parallel implementation.
- Because the system requires no object matching, it may be able to detect moving objects that are camouflaged or partially obscured.

The limitations of the technique are:

- It requires the location of the FOE. However, this may be a fundamental requirement for systems that are to reliably detect moving objects from a moving platform.
- The method used to detect vertical motion is most effective when detecting vertical movement of horizontal (or near-horizontal) edges in the CLM image. It requires that moving objects contain such edges.
- The system ignores motion that is *exactly* along lines radiating from the FOE, but such motion appears to be rare. Note that the truck in the road scene was traveling rough-



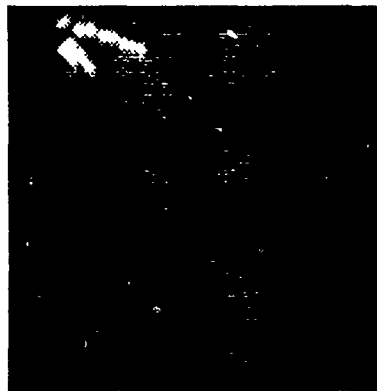
(a) CLM image convolved with horizontal edge mask of Sobel operator.



(b) Map of horizontal edge components exhibiting vertical motion.



(c) Thresholded map depicting detected motion.



(d) Detected motion transformed into rectangular reference frame.



(e) Resulting motion overlaid upon dimmed version of original image.

Figure 5: Intermediate and final results of detecting vertical motion in the CLM reference frame.

ly along lines radiating from the FOE, but was still detected.

8 SIMILARITIES TO BIOLOGICAL SYSTEMS

Two important similarities exist between biological vision systems and the techniques presented here for motion detection. First, the CLM approximates the mapping found between cells of the retina and the visual cortex [Schwartz, 1980].

Second, areas have been found in the visual cortex that specifically detect the motion of edges as described in [Hubel, 1981]. However, the moving-edge detector described here differs from those observed in biological systems in the following way. In the visual cortex, moving-edge detectors are sensitive to the *sense* of movement of an edge. That is, if a horizontal edge moves upward, then one set of detectors is triggered. If the edge moves downward, then a different set of detectors is triggered. The detector presented here has no such sensitivity, and responds equally to both directions of movement.

One possible use of such detectors in biological systems is to detect moving objects when the observer is moving. Also,

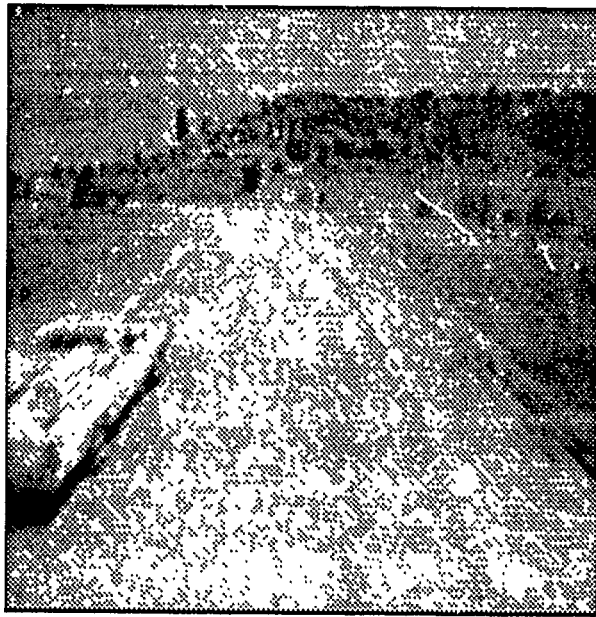
by adding a mechanism to detect horizontal movement (instead of just vertical movement) in the CLM reference frame, we can produce a motion detector suitable for a stationary observer.

9 FUTURE WORK

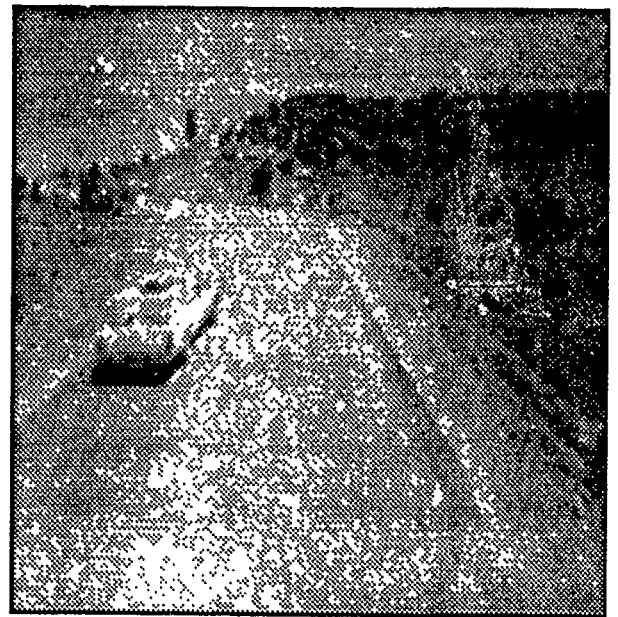
The next step in development of the system will be to determine requirements for computing the FOE. It may be possible to simply use an FOE computed from the orientation of the camera with respect to the vehicle.

We are also exploring a Hough-transform based method for determining the paths of edges within the scene over time. When plotted as x-location (y-location) versus time, vertical (horizontal) edges associated with stationary objects form hyperbolas (assuming only translational observer motion). All such hyperbolas share a common asymptote, which can be detected by the Hough transform, and which is the FOE. Also note that the parameters of such hyperbolas should be sufficient for detecting potential collisions with stationary objects via looming information as described in [Arbib, 1989].

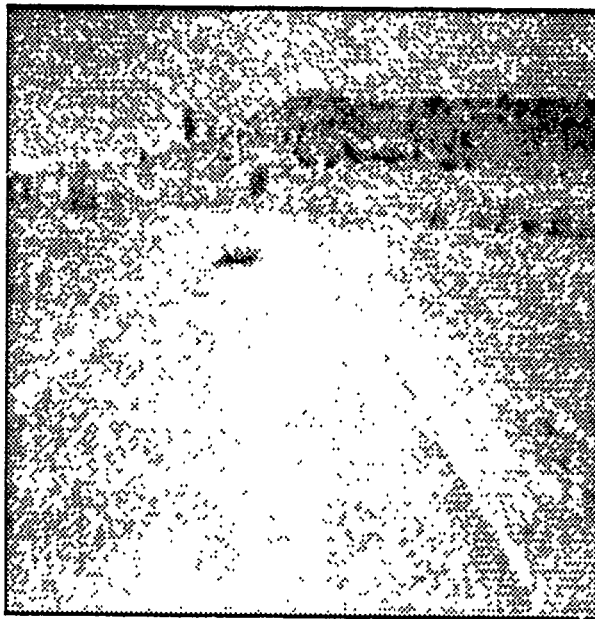
We will study automating the setting of the movement threshold, based on image statistics of the horizontal Sobel



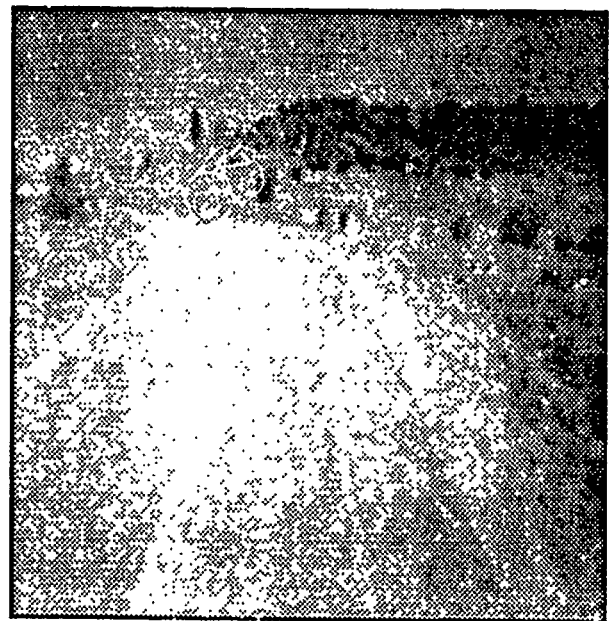
(a) Fourth frame.



(b) Fifth frame.



(c) Eighth frame.



(d) Fifteenth frame.

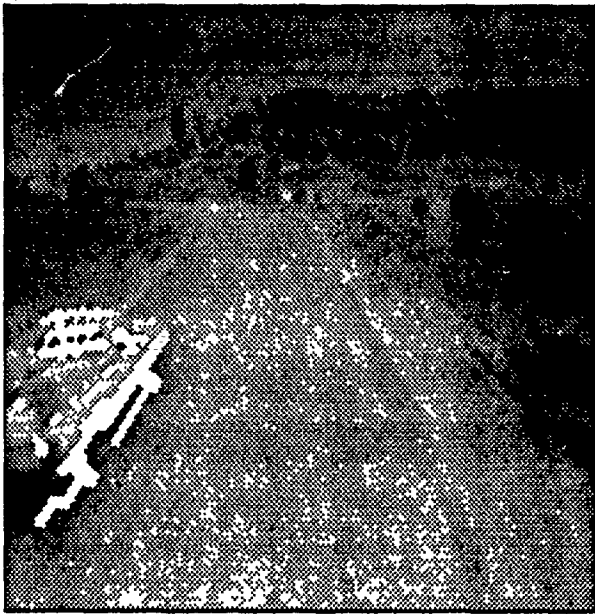
Figure 6: Real imagery of a moving truck, taken from a moving observation vehicle following a road.

image and on the desired distance (in pixels) of vertical movement above which motion is to be detected.

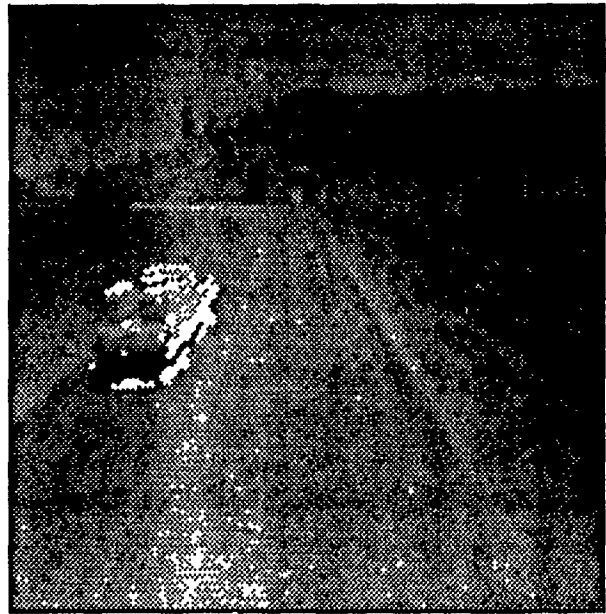
Segmentation of the moving object will be investigated. Initial experimental results indicate that, once a moving object is detected, a rough segmentation (or separation of the object from its background) may be performed by reducing the movement threshold in the neighborhood of the object.

This preliminary segmentation can then be passed to other processes for more detailed analysis.

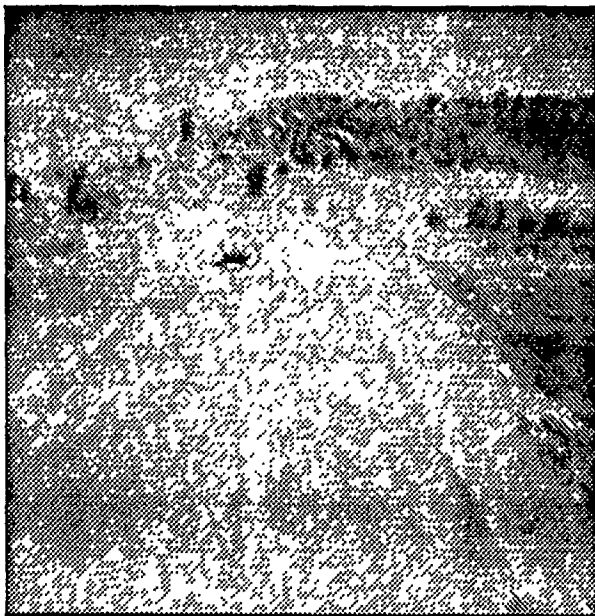
Finally, merging of system results over time may provide more accurate information about detected moving objects, and will be examined.



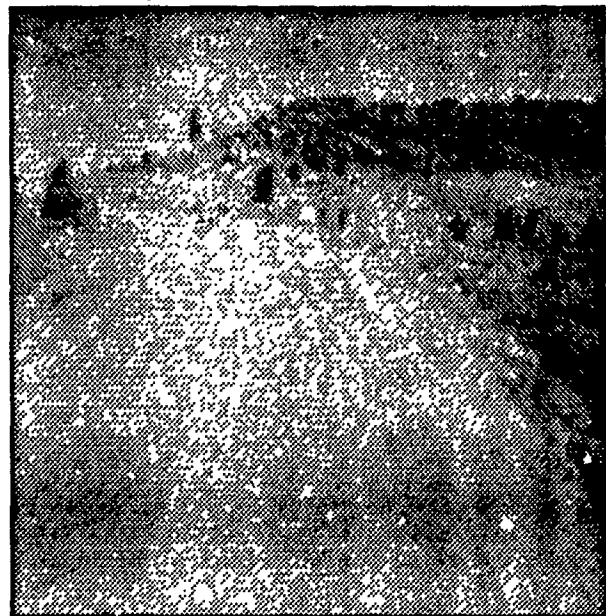
(a) Fourth frame.



(b) Fifth frame.



(c) Eighth frame.



(d) Fifteenth frame.

Figure 7: Detected motion overlaid upon dimmed versions of images from the real truck sequence.

10 CONCLUSION

We have presented a system to detect edge segments of moving objects viewed by a translating camera. The system uses two key concepts. First, a Complex Logarithmic Mapping is performed to simplify the process of detecting the desired movement. Second, a mechanism to detect moving edges has

been developed to detect vertical motion within the CLM image.

The system has been tested on a sequence of real images in which it successfully detects, with no false alarms, portions of a moving vehicle that is viewed by a translating observer. The system does not require densely-sampled imagery, it detects moving objects of various sizes, and tolerates gradual change-

es in camera orientation. It currently requires only one parameter whose value is not critical, it is computationally efficient, highly suited to parallel processing, and may be able to detect moving objects that are camouflaged or partially obscured.

We plan to extend the system to detect complete objects (instead of portions of objects), and to automatically compute the FOE.

References

- [Arbib, 1989] M. A. Arbib, *The Metaphorical Brain*, pages 333-8, 1989.
- [Bhanu and Burger, 1988] B. Bhanu, W. Burger. Qualitative motion detection and tracking of targets from a mobile platform. *Proceedings of the DARPA Image Understanding Workshop*, pages 289-313, Cambridge, Massachusetts, 1988.
- [Blostein and Huang, 1988] S.D. Blostein, T.S. Huang. A tree search algorithm for target detection in image sequences. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 832-7, Ann Arbor, Michigan, 1988.
- [Burger and Bhanu, 1989] W. Burger, B. Bhanu. On computing a "fuzzy" focus of expansion for autonomous navigation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 563-8, San Diego, California, 1989.
- [Heeger and Hager, 1988] D.J. Heeger, G. Hager. Egomotion and the stabilized world. *Proceedings of the Second International Conference on Computer Vision*, pages 435-40, 1988.
- [Hubel, 1981] D. H. Hubel. Evolution of ideas on the primary visual cortex, 1955-1978: a biased historical account. *Nobel Lecture*, 1981.
- [Jain and O'Brien, 1984] R. Jain, N. O'Brien. Ego-motion complex logarithmic mapping. *Proc. SPIE Int. Soc. Opt. Eng.*, 521:16-23, 1984.
- [Jain, 1984] R. Jain. Segmentation of frame sequences obtained by a moving observer. *Pattern Analysis and Machine Intelligence*, 6:624-629, 1984.
- [Kahn, 1988] P. Kahn. Integrating moving edge information along a 2D trajectory in densely sampled imagery. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 702-9, Ann Arbor, Michigan, 1988.
- [Lee and Lin, 1988] J.S.J. Lee, C. Lin. A novel approach to real-time motion detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 730-5, Ann Arbor, Michigan, 1988.
- [Schwartz, 1980] E.L. Schwartz. Computational anatomy and functional architecture of striate cortex: a spatial mapping approach to coding. *Vision Research*, 20:645-669, 1980.
- [Thompson and Pong, 1990] W.B. Thompson and T.C. Pong. Detecting moving objects. *International Journal of Computer Vision*, 4:39-57, 1990.
- [Zhu, 1988] Q. Zhu. Structural pyramids for representing and locating moving obstacles in visual guidance of navigation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 832-7, Ann Arbor, Michigan, 1988.

Acknowledgments

James Frazier was supported by the Hughes Aircraft Company Doctoral Fellowship Program. Ramakant Nevatia and this research were supported by the Defense Advanced Research Projects Agency under Contract # F49620-89-C-0126 and monitored by the Air Force Office of Scientific Research.

Motion and Binocular Stereo Integrated System for Passive Ranging

Peter F. Symosek, Bir Bhanu,
Scott Snyder and Barry Roberts
Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

ABSTRACT

Range measurements to objects in the world relative to mobile platforms such as ground or air vehicles are critical for visually aided navigation and obstacle detection/avoidance. Active (laser) range sensors can be used to provide such range measurements although they have a limited field of view, suffer from slow data acquisition, and are expensive. This paper reports the results obtained to date for a robust and efficient *passive* technique for obtaining range measurements. Our approach consists of a unique and synergistic combination of two types of passive ranging: binocular stereo and motion stereo. The problem that we address is the optimal combination of sparse motion stereo range estimates, $r_m(x,y)$, and sparse binocular stereo range estimates, $r_s(x,y)$, so the resulting range map is as accurate and dense as possible throughout the entire field of view.

1. INTRODUCTION

The objective of this research is to develop a passive ranging system that reaps the benefits of binocular and motion stereo and requires only two cameras. The innovative stereo system synergistically combines binocular and motion stereo modalities by combining of interest point matching, range measurement blending and Kalman filters.

The important benefits of the proposed hybrid system are:

1. The system is cheap to build (compared to active sensors).
2. It is passive (i.e. non-detectable, covert).
3. A more dense and more accurate range map is generated than is generated by either passive technique alone which is necessary for obstacle avoidance.
4. Negligible motion distortion is caused by the moving platform (i.e. fast data acquisition).

Previous efforts in the derivation of approaches for the synergistic combination of binocular and motion stereo ranging have placed restraints on the problem specification to reduce the complexity of the analysis. To date, no demonstration of a totally general, comprehensive characterization of the ranging problem for multiple binocular stereo frames has been derived. The purpose of the technique which this paper discusses is the empirical evaluation of the performance and robustness of such a system for various scenarios.

Features from temporally-distinct stereo pairs of images can be matched with greater reliability to improve the accuracy of disparity-based range calculations. Leung, et al¹ derived an algorithm for finding point correspondences among stereo image pairs at two consecutive time instants (t_{i-1} , t_i). They demonstrated significantly improved feature matching accuracy for scenes that demonstrate large feature displace-

ments due to object motion in the scene. Li and Duncan² estimated the platform motion from measures for the optical flow of the left and right cameras for a series of binocular stereo images, without point-to-point correspondences. In addition, stereo matching procedures based on the estimated translational velocity and the flow fields were derived. An empirical evaluation of the robustness of the approach to image noise (which degrades the accuracy of the flow field) for synthetic images was carried out. The approach was demonstrated to be robust for representative flow field magnitude and direction errors. Binocular and motion stereo ranging techniques attain distinct degrees of accuracy for various image regions. The regions of good and poor performance of each approach was derived by Sridhar and Suorsa³ for binocular images obtained with a computer controlled motion table. These images were a scaled representation of the imagery that would be obtained by a helicopter traveling at a speed of 20 knots. The image processing rate was 2 frames/sec. The scaled stereo baseline was .1 inches which corresponds to a true baseline of 5 m. They demonstrated a "recursive motion algorithm", based on an extended Kalman filter aided by an inertial navigation system, which produced good results for scene objects whose images were not near the image location corresponding to the instantaneous direction of travel. They also demonstrated a "recursive stereo algorithm", based on geometric disparity, which produced good results for all portions of the image. The binocular and motion stereo algorithms were not integrated and no method for estimation of confidence factors for each measurement approach was presented.

The following tasks were carried out to demonstrate the motion and binocular stereo integrated system:

1. Laboratory collection of binocular and motion stereo data (a total of 5 pair of frames). For use in validating our integrated stereo technique, ground truth range to a selected number of image points was obtained.
2. Development and software implementation of binocular and motion stereo algorithms and the integration of the two algorithm suites. This includes the extraction of "interest" points and the matching of interest points for subsequent binocular and motion stereo range computations.
3. Modeling of errors in binocular and motion stereo. The functional form of these equations is given in a subsequent section.⁴
4. Kalman filter definition and implementation. We developed Kalman filter software to process range data measurements and derive estimates for the error states which contribute to range error.
5. Evaluation of the integrated stereo system with real imagery.

The next section of this paper presents the motion and binocular stereo integrated system technical approach. Section 3 discusses results obtained during an empirical evaluation of the performance of the system with the laboratory data and simulated Inertial Reference Unit (IRU) data. The last section of the paper discusses our plans for future research to further optimize the system.

2.0 TECHNICAL APPROACH

With a two camera system in motion, a stereo ranging system is formed which consists of *binocular stereo* and *motion stereo* range computations. In the case of binocular stereo two cameras are rigidly mounted on the same fixture such that their optical axes are parallel and yet horizontally displaced by a fixed, known distance. Whereas the cameras are laterally displaced for binocular stereo, the cameras are longitudinally displaced, due to forward vehicle motion, for motion stereo. On a moving platform, the same two cameras can provide the imagery required to perform one binocular and two motion stereo range calculations.

Tradeoffs between binocular and motion stereo are shown in Figure 1. Binocular stereo and motion stereo compute range to "distinguished" points in the image. As shown in Figure 1, binocular stereo range computations suffer the greatest error at the edges of the camera's field of view (FOV), where motion stereo range is most accurate. The converse error relationship holds true in the vicinity of the focus of expansion where motion stereo range error is great and binocular stereo range error is small.

Our integrated stereo system shown in Figure 2 uses two key elements which constitute the unique features of our approach:

- (1) matching of interesting points in binocular stereo and motion stereo imagery,
- (2) modeling of range errors present in the motion and binocular stereo techniques. These errors determine the state vector in the Kalman filter application used to obtain improved estimates of range values.

The coincident points of interest, i.e. those points for which range is computed by both motion and binocular stereo techniques, are used as measurements to estimate errors in the ranging processes. The points in the range maps which are not coincident can be corrected with these error estimates, improving the overall quality of the composite range map. This can be achieved with the use of a blending filter as shown Figure 3. This filter derives a composite range map for each measurement location as the weighted average of the Kalman filter estimates for the range, where the averaging weights are the current estimates of the measurement noise obtained from each filter. The confidence in each range measurement is inversely proportional to the estimate of the measurement noise, so that when the measurement noise for the binocular stereo algorithm is large, the estimate obtained from the motion stereo Kalman filter is weighted more heavily. Conversely, when the measurement noise for the motion stereo algorithm is large, the estimate from the binocular stereo algorithm is given more weight. The filtered estimates of the measurement noise are used to smooth out the effects of isolated bad measurements.

2.1 Range Error Modeling Results

We derived the functional relationships of errors in range values.

The total differential of motion stereo range:

$$dR_f = \frac{\partial R_f}{\partial y'} dy' + \frac{\partial R_f}{\partial z'} dz' + \frac{\partial R_f}{\partial y} dy + \frac{\partial R_f}{\partial z} dz + \quad (1)$$

$$\frac{\partial R_f}{\partial f_{ov_h}} df_{ov_h} + \frac{\partial R_f}{\partial f_{ov_v}} df_{ov_v} + \frac{\partial R_f}{\partial F} dF +$$

$$\frac{\partial R_f}{\partial \Delta\psi'} d\Delta\psi' + \frac{\partial R_f}{\partial \Delta\theta'} d\Delta\theta' + \frac{\partial R_f}{\partial \Delta\phi'} d\Delta\phi' +$$

$$\frac{\partial R_f}{\partial v_x} dv_x + \frac{\partial R_f}{\partial v_y} dv_y + \frac{\partial R_f}{\partial v_z} dv_z$$

where,

(y', z') = pixel location of an interest point in the left frame of a motion stereo pair of images that is acquired at time t_{i+1} .

(y, z) = pixel location of the interest point in the left frame of a motion stereo pair of images that is acquired at time t_i and matches (y', z') .

f_{ov_v} = camera vertical field-of-view.

f_{ov_h} = camera horizontal field-of-view.

$\Delta\psi'$ = change in yaw angle that occurred in the time interval $t_{i+1}-t_i$.

$\Delta\theta'$ = change in pitch angle that occurred in the time interval $t_{i+1}-t_i$.

$\Delta\phi'$ = change in roll angle that occurred in the time interval $t_{i+1}-t_i$.

(v_x, v_y, v_z) = the velocity of the camera.

F = the focal plane to lens center distance.

The total differential of binocular stereo range:

$$dR_f = \frac{\partial R_f}{\partial y_l} dy_l + \frac{\partial R_f}{\partial z_l} dz_l + \frac{\partial R_f}{\partial y_r} dy_r + \frac{\partial R_f}{\partial z_r} dz_r + \quad (2)$$

$$\frac{\partial R_f}{\partial f_{ov_h}} df_{ov_h} + \frac{\partial R_f}{\partial f_{ov_v}} df_{ov_v} + \frac{\partial R_f}{\partial \Delta\psi} d\Delta\psi + \frac{\partial R_f}{\partial \Delta\theta} d\Delta\theta +$$

$$\frac{\partial R_f}{\partial \Delta\phi} d\Delta\phi + \frac{\partial R_f}{\partial F} dF + \frac{\partial R_f}{\partial a} da$$

where,

(y_l, z_l) = pixel location of an interest point in the left frame of a binocular stereo pair of images acquired at time t_i .

(y_r, z_r) = pixel location of an interest point in the left frame of a binocular stereo pair of images acquired at time t_i and matches (y_l, z_l) .

$\Delta\psi$ = the boresight yaw angle.

$\Delta\theta$ = the boresight pitch angle.

$\Delta\phi$ = the boresight roll angle.

a = camera separation distance.

We also derived the functional relationships between the variance of range error and the location of an interest point in the field of view.

An approximation to the range calculation error for the case of motion stereo range computations is described as

$$\sigma_M(u_A, v_A) = \sigma_{D_M} \frac{\Delta R_M(u_A, v_A)}{\sqrt{F^2 + u_A^2 + v_A^2}}, \quad (3)$$

where

σ_{D_M} = an initial estimate of the range calculation error due to the error in motion stereo point matching algorithm.

$\Delta R_M(u_A, v_A)$ = is the computed error in range for the world point whose projection onto the image plane is described in three space by (F, u_A, v_A) .

F = is the distance between the lens center and the image plane.

Likewise, an approximation to the range calculation error for the case of binocular stereo range computations is described here,

$$\sigma_B(u_L, v_L) = \sigma_{D_B} \frac{\Delta R_B(u_L, v_L)}{\sqrt{F^2 + u_L^2 + v_L^2}}, \quad (4)$$

where

σ_{D_B} = an initial estimate of the range calculation error due to the error in the binocular stereo point matching algorithm.

$\Delta R_B(u_L, v_L)$ = is the computed error in range for the world point whose projection onto the image plane is described in three space by (F, u_L, v_L) .

In computing range with either the motion stereo or binocular stereo techniques, all range measurements are made relative to the first of a temporal pair of images (i.e. A of A and B images) and the left image of a stereo pair, as shown in Figure 4. Hence the subscripts A and L are used for the variables that describe points in three space on the image plane. In our implementation, the A and L images are the same image.

2.2 Kalman Filter Implementation

The twenty-nine error states summarized in Table 1 are mechanized in the Kalman Filter. The first 7 states are based on the level axis "PSI-Angle" IRU error model developed by M. Ignagni.⁵

$$\dot{\underline{\Psi}} = -(\underline{\rho} + \underline{\Omega}) \underline{\Psi} - C \delta \underline{\omega} \quad (5)$$

$$\delta \dot{\underline{V}} = C \delta \underline{A}^B - \underline{\Psi} \underline{X} \underline{A}^L (\delta \underline{R} \cdot \underline{R}/R) (\underline{R}/R) + \delta \underline{g}' \quad (6)$$

$$\delta \dot{\underline{R}} = \delta \underline{V} - \underline{\rho} \underline{X} \delta \underline{R} \quad (7)$$

where:

$\underline{\Psi}$ = Psi-angle error (states 1, 2, and 3).

$\delta \underline{V}$ = Psi-angle horizontal velocity error (states 4 and 5).

$\delta \underline{R}$ = Psi-angle horizontal position error (states 6 and 7).

$\underline{\rho}$ = local level transport rotation rate (V/R).

$\underline{\Omega}$ = Earth rate in local level coordinate frame.

C = Body to Local Level Direction Cosine Transformation Matrix

$\delta \underline{\omega}$ = Gyro error states (states 25, 26, 27).

$\delta \underline{A}^B$ = Accelerometer error states (states 28 and 29).

\underline{A}^L = Local level acceleration.

ω_s = Shuler frequency (~0.00125 rps).

\underline{R}/R = unit vector.

$\delta \underline{g}'$ = gravity deflection and anomaly errors.

The vertical error states (8, 9, 10) assume an IRU vertical channel damped with a reference altitude. Figure 5 shows a typical IRU vertical channel filter. The error model implemented in the Kalman Filter can be expressed as:

$$\dot{x}_8 = -x_9 \quad (8)$$

$$\dot{x}_9 = K_1 x_9 + x_{10} \quad (9)$$

$$\dot{x}_{10} = x_{24} + K_2 x_9 - K_3 x_8 \quad (10)$$

where K_1, K_2, K_3 , are the vertical channel gains. These gains were selected as 0.6, 0.15, and 0.0156, respectively.

The remaining error states are modeled as Gauss-Markov Processes with large time constants. The large time constants effectively model the error sources as constants. A Gauss-Markov process can be represented as:

$$\dot{x} = \frac{-1}{\tau} x + \eta \quad (11)$$

where η is a white noise process and τ is the time constant.

The binocular stereo and motion stereo range errors are not states within the Kalman Filter. They are linear combinations of the Kalman Filter error states. This linear combination can be expressed as:

$$\delta R_s = H \hat{x} \quad (12)$$

$$\delta R_m = H \hat{x} \quad (13)$$

where H is the measurement matrix defined by the equations given earlier and \hat{x} is the estimated error state vector. Therefore to calculate improved binocular stereo and motion stereo ranges, the input values to the binocular and motion algorithms would be modified by their error values and rerun. Alternatively the H -matrices for each binocular or motion stereo range point could be formulated and the ranges corrected for the linear combination of error states. The latter is done in the current implementation of the binocular and motion stereo integrated stereo.

The filter's binocular stereo measurement consists of motion stereo range subtracted from binocular stereo range. The filter's motion stereo measurement is the negative of the filter's binocular stereo measurement. The measurement process is modeled as follows:

$$y_{mj}(k) = R_{mj} - R_{sj} = H x_m + v_m \quad (14)$$

$$y_{sj}(k) = R_{sj} - R_{mj} = H x_s + v_s \quad (15)$$

where,

$y_{mj}(k)$ is the measurement for the motion stereo Kalman filter for the j 'th feature point location at time k ,

$y_{sj}(k)$ is the measurement for the binocular stereo Kalman filter for the j 'th feature point location at time k ,

R_{mj} is the estimate of the range to the three-dimensional location from the motion stereo algorithm,

R_{sj} is the estimate of the range to the three-dimensional location from the binocular stereo algorithm,

v_m is measurement noise, $E\{v_m^T v_m\} = \sigma_M^2$, small near focus of expansion (FOE), large near periphery and

v_s is measurement noise, $E\{v_s^T v_s\} = \sigma_B^2$.

The noise variances σ_M^2 and σ_B^2 are calculated with equations (3) and (4).

3.0 EXPERIMENTS

For experimentation with the system, we modified existing Kalman Filter software previously used in a real-time environment. The software is written in C and FORTRAN and was modified for a simulation environment. The PSI-angle IRU error model was already implemented in the software. We added states 8 through 29, modified the routine which calculates the H-matrix, and wrote new input/output routines.

For the initial evaluation phase, our stereo system is not fully integrated, but left in modular components. These components consist of laboratory collected video files, an IRU error model simulation, binocular stereo range algorithm, motion stereo range algorithm and Kalman Filter algorithm. Each of these components are run separately with communication between the components through input/output files.

Five frames of video data were collected in the laboratory at 2 foot intervals. An example of the experimental data is shown in Figure 6. From these five frames the binocular stereo ranges are calculated to various points with the binocular stereo range software.

To simulate motion for the motion stereo algorithm we chose two velocities, 2 ft/sec and 20 ft/sec. These two velocities correspond to processing the four frames at 1 second intervals or 0.1 second intervals. The attitude of both experiments was chosen to be level and in a northerly direction. A detailed description of the motion stereo algorithm is presented in the paper by Bhanu and Roberts.⁴

IRU errors were simulated by running an off-line IRU error simulation and adding the errors onto our nominal motion. The simulation used is commonly called the HINS simulation and is a monte-carlo simulation of the IRU error equations. For this research, we simulated a GG1328 gyro based IRU.

The trajectory chosen was a northern cruise at 15 ft/sec. Figure 7 shows simulation results for the first 10 seconds. For a cruise scenario such as the trajectory above, IRU errors are essentially a function of time, therefore, to formulate IRU errors for our two multieye stereo cases the true trajectory was subtracted from the Figure 7 data. The resultant error data was then added to our multieye stereo trajectory to simulate corrupted IRU data.

The output binocular stereo and motion stereo range files, and simulated IRU data files are read into the Kalman Filter software. The filter software runs a range matching algorithm to detect coincident range points. For each coincident point the corresponding H-matrix and filter measurements are calculated and processed by the filter.

Results from processing the first frame of the five frame sequence with the Kalman Filter is shown in Table 2. These results are with simulated IRU noise and a 1 Hz video frame iteration rate, which equates to a 2 ft/sec velocity. Groundtruth measurements for the 12 matched feature point locations of frame 1 are presented in Table 3.

The processing of the first frame does show promise. As shown for measurement 1, the corrections added to the binocular stereo range and motion stereo range tend to converge the solutions to a common point as expected. In general this behavior can be observed in the results for measurements 1 through 12. There are some exceptions (measurement 8 and 10) which could possibly be due to the measurement weighting.

The 20 ft/sec velocity case (.1 Hz video frame iteration rate) provided results similar to the 2 ft/sec case. Table 4 contains filter output from processing the first and second frames through the Kalman Filter. Results for Frame 1 processing match the results presented in Table 2. Frame 2 which is a propagation in time shows favorable results with the exception of measurements 8 and 10. The filter has propagated forward a large binocular stereo range error which is not reflected in measurement 8 of frame 2. Groundtruth measurements for the 13 matched feature point locations of frame 1 and the 13 matched feature point locations of frame 2 are presented in Table 5.

4. CONCLUSIONS

We presented the basic concept and initial results of our binocular and motion stereo integrated system. The basic software is in place with the exception of the blending filter and work at improving its performance is continuing. Our current iterations do show a lot of promise and exhibit expected behavior. Thorough evaluation of the system's performance for large amounts of data and multiple scenarios will be reported in the future.

5. REFERENCES

1. M. K. Leung, A. N. Choudhary, J. H. Patel, and T. S. Huang, *Point Matching in a Time Sequence of Stereo Image Pairs and its Parallel Implementation on a Multiprocessor*, Proceedings of the Workshop on Visual Motion (March 20-22, 1989).
2. L. Li and J. H. Duncan, *Recovering 3-D Translational Motion and Establishing Stereo Correspondence from Binocular Image Flows*, Proceedings of the Workshop on Visual Motion (March 20-22, 1989).
3. B. Sridhar and R. Suorsa, *Integration of Motion and Stereo Sensors in Passive Ranging Systems*, Proceedings of the American Control Conference (1990).
4. B. Bhanu and B. Roberts, *INS-Integrated Motion Analysis for Autonomous Vehicle Navigation*, Proceedings of the DARPA Image Understanding Workshop (1990).
5. M. Ignagni, "Error Analysis of a Strapdown Terrestrial Navigation System," Internal Honeywell Document (1989).

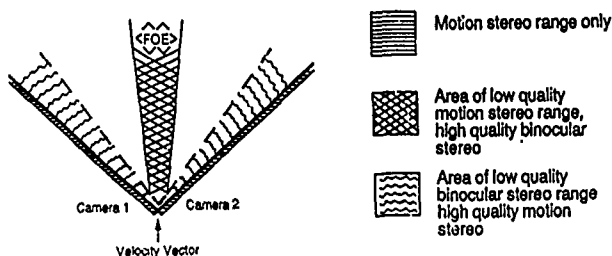


Figure 1. Modalities for passive ranging

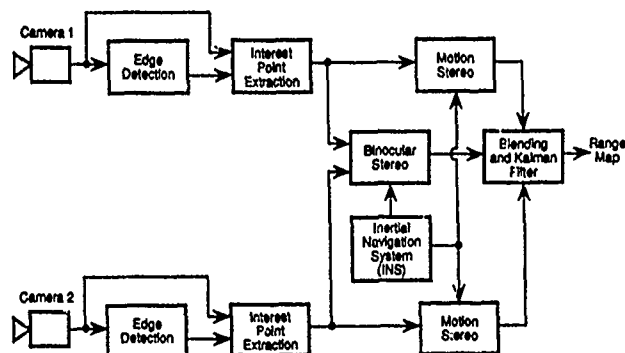


Figure 2. System for the integration of motion stereo and binocular stereo

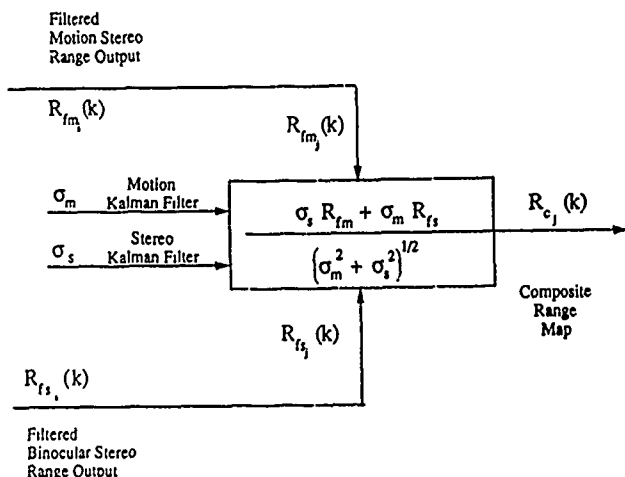


Figure 3. Composite Range Map/Blending Filter

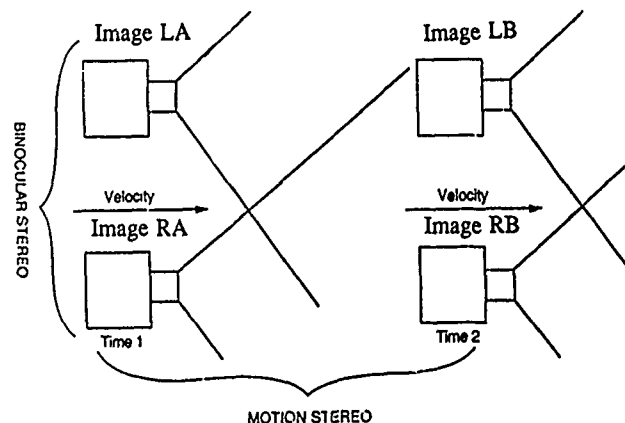


Figure 4. Motion and binocular stereo data collected

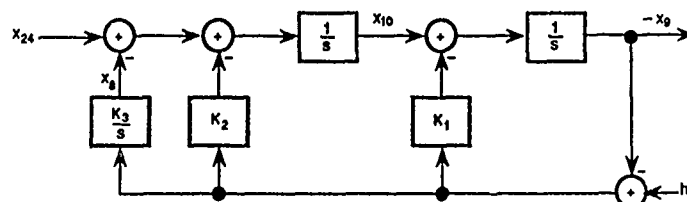


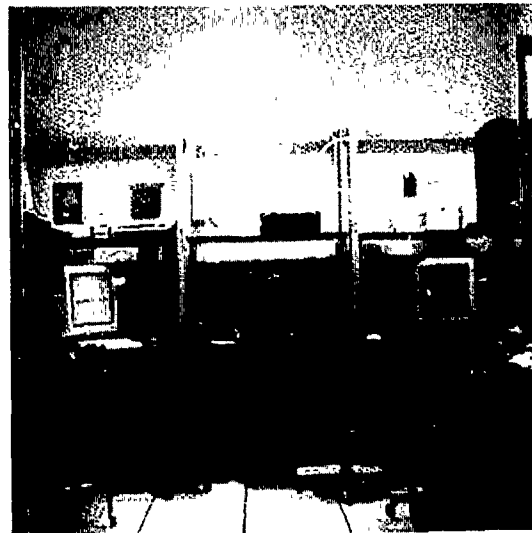
Figure 5. Block diagram of vertical channel filter

Table 1. Error states used in Kalman filtering

- | | |
|--|--|
| 1. IRU psi 1 angle error | 16. y_r right camera Y optical axis offset error |
| 2. IRU psi 2 angle error | 17. z_r right camera Z optical axis offset error |
| 3. IRU psi 3 angle error | 18. Camera yaw angle boresight error |
| 4. IRU x velocity error | 19. Camera pitch angle boresight error |
| 5. IRU y velocity error | 20. Camera roll angle boresight error |
| 6. IRU x position error | 21. Camera separation distance (a) |
| 7. IRU y position error | 22. y_l' left camera Y optical axis offset error (Frame B) |
| 8. Vertical channel acceleration error | 23. z_l' left camera Z optical axis offset error (Frame B) |
| 9. Vertical channel velocity error | 24. Z accelerometer bias |
| 10. Vertical channel position error | 25. X gyro bias error |
| 11. Horizontal FOV error (fov_h) | 26. Y gyro bias error |
| 12. Vertical FOV error (fov_v) | 27. Z gyro bias error |
| 13. Camera focal plane to lens center distance (F) | 28. X accelerometer bias error |
| 14. y_l left camera Y optical axis offset error | 29. Y accelerometer bias error |
| 15. z_l left camera Z optical axis offset error | |



(a)



(b)



(c)

Figure 6. Laboratory image database, frame 1. (a) Frame 1 image obtained from left camera of stereo pair, (b) Frame 1 image obtained from right camera of stereo pair, (c) Frame 2 image obtained from left camera of stereo pair.

Table 2. KF Computed Range Errors One Hertz Processing

Simulated IRU Errors, 2 FPS Velocity						
CAMERA PARAMETERS						
hfov = 0.754160 vfov = 0.313147 F = 0.041000 a = 2.000000						
Time = 1.0 second (Frame 1)						
measurement	raw binocular range	raw motion range	KF binocular error	KF motion error	corrected binocular range	corrected motion range
1	23.470947	13.229049	9.257078	-1.906973	14.213869	15.136022
2	15.250125	19.987539	-2.715582	-1.591671	17.965708	21.579210
3	23.286278	11.497955	5.687111	-4.775131	17.599167	16.273087
4	17.710770	22.075123	-1.643602	1.591987	19.354372	20.483137
5	13.850588	20.908190	-3.369545	7.472088	17.220133	13.436102
6	15.973729	21.540310	-3.376971	2.922868	19.350700	18.617441
7	16.092087	17.760782	-1.714687	2.406360	17.806774	15.354422
8	16.151932	14.471107	-2.965745	2.350610	19.117678	12.120497
9	21.183895	22.302511	2.563853	2.367056	18.620041	19.935455
10	15.358275	14.538172	-3.013412	1.480840	18.371687	13.057332
11	18.167021	20.215263	0.081987	1.600948	18.085033	18.614315
12	15.797955	21.457747	-1.679191	1.594449	17.477146	19.863297

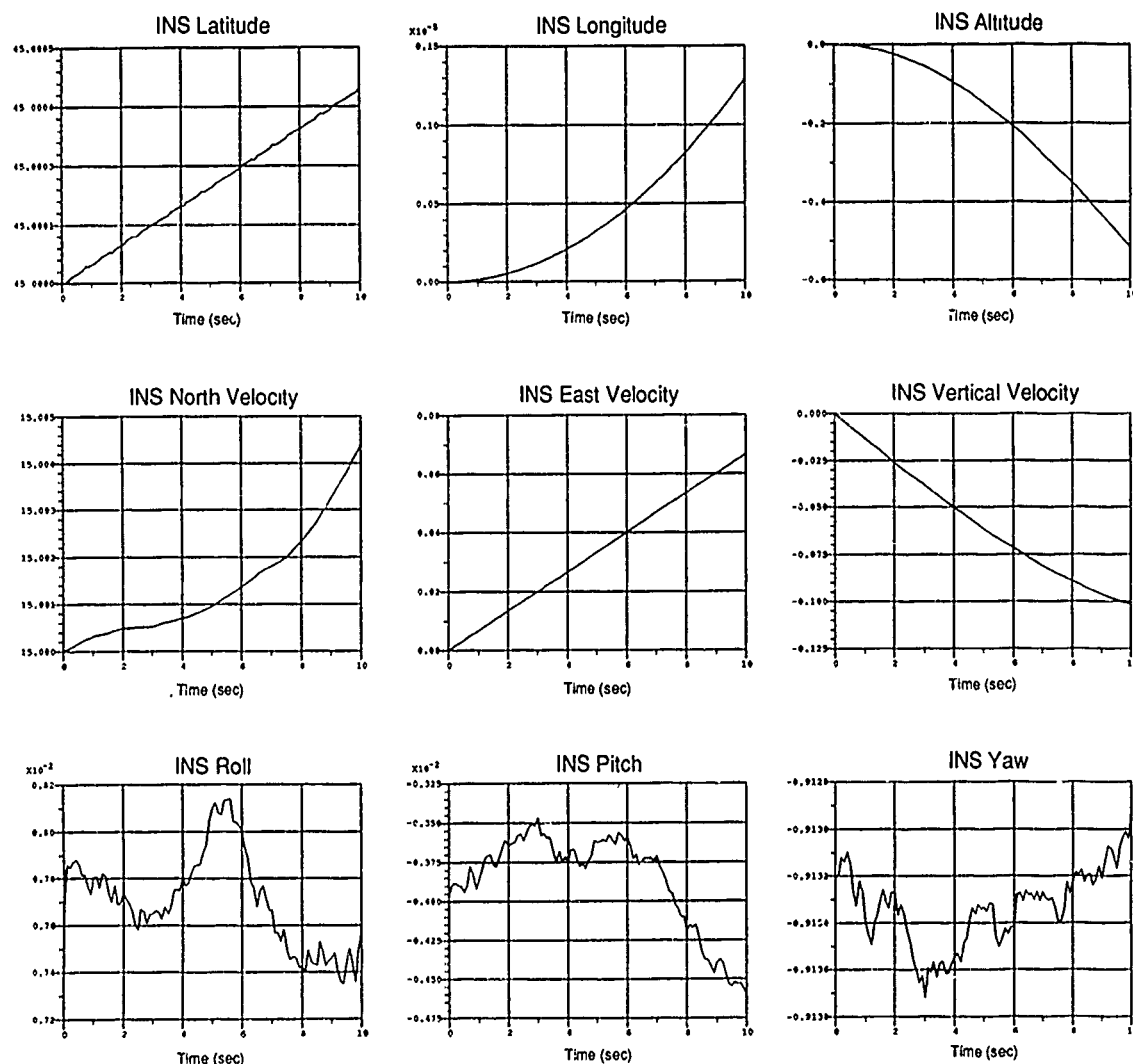


Figure 7. Simulation of a GG1328 gyro based IRU

measurement	y_l [pixels]	z_l [pixels]	y_r [pixels]	z_r [pixels]	y_l [pixels]	z_l [pixels]	R_f [feet]
Time = 1.0 second (Frame 1)							
1	-152	-59	-182	-53	-128	-51	13.9764
2	-52	122	-131	114	-46	111	18.8474
3	-41	-50	-94	-38	-38	-43	12.2229
4	-12	113	-83	106	-9	104	19.1300
5	41	65	-58	57	37	60	19.2826
6	49	202	-39	189	44	184	20.2861
7	92	-61	0	-49	82	-53	18.0941
8	95	167	0	149	82	144	14.0358
9	124	-15	51	-8	113	-12	22.3442
10	157	152	48	143	134	138	14.2818
11	170	-8	81	-1	153	-5	20.1257
12	170	-16	71	-9	154	-12	21.4648

Table 4. KF Computed Range Errors 0.1 Hertz Processing

Simulated IRU Errors, 20 FPS Velocity						
CAMERA PARAMETERS						
hfov = 0.754160 vfov = 0.313147 F = 0.041000 a = 2.000000						
measurement	raw binocular range	raw motion range	KF binocular error	KF motion error	corrected binocular range	corrected motion range
Time = 0.2 (Frame 1)						
1	23.470947	12.980159	9.617279	-1.099180	13.853668	14.079339
2	15.250125	19.459459	-2.733061	0.434156	17.983187	19.025303
3	23.286278	10.937357	5.861856	-2.925393	17.424423	13.862750
4	15.922381	14.836158	-1.956218	-4.532531	17.878599	19.368689
5	17.710770	22.263815	-1.671963	2.354116	19.382732	19.909698
6	13.850588	21.918085	-3.384527	9.700621	17.235115	12.217464
7	15.973729	21.828583	-3.465710	6.287485	19.439438	15.541098
8	16.092087	18.314566	-1.659888	0.328105	17.751974	17.986462
9	16.151932	14.771038	-3.026224	2.168971	19.178156	12.602067
10	21.183895	22.844698	2.680561	1.348993	18.503334	21.495705
11	15.358275	14.761926	-3.051116	0.234672	18.409391	14.527253
12	18.167021	20.570257	0.180036	0.792287	17.986984	19.777969
13	15.797955	21.834696	-1.613573	0.830073	17.411528	21.004623

Time = 0.3 (Frame 2)

1	15.395482	15.375104	-2.823521	-1.199930	18.219004	16.575033
2	15.055490	18.953318	-3.251050	-3.732748	18.306541	22.686066
3	15.658957	19.364252	-2.859818	-3.794225	18.518774	23.158478
4	14.771465	16.280893	-3.150485	-4.380288	17.921949	20.661182
5	16.188807	18.075811	-3.643293	6.040530	19.832100	12.035282
6	16.690779	18.752043	-2.562827	4.724038	19.253607	14.028005
7	15.174622	18.209389	-3.081228	3.214266	18.255850	14.995123
8	19.333355	24.704124	-1.267558	6.274734	20.600912	18.429390
9	14.181705	17.482733	-3.954892	1.894178	18.136597	15.588555
10	16.496565	16.344479	-3.117268	0.850061	19.613832	15.494417
11	14.511797	16.740181	-4.223449	1.269790	18.735247	15.470390
12	15.358172	18.337030	-2.874749	2.921894	18.232922	15.415136
13	13.100904	19.790014	-4.306006	2.210015	17.406910	17.580000

Table 5. Groundtruth Measurements for .1 Hertz Processing

measurement	y_l [pixels]	z_l [pixels]	y_r [pixels]	z_r [pixels]	y_l [pixels]	z_l [pixels]	R_f [feet]
Time = 0.2 (Frame 1)							
1	-152	-59	-182	-53	-128	-51	13.9764
2	-52	122	-131	114	-46	111	18.8474
3	-47	-50	-94	-38	-38	-43	12.2229
4	-13	0	-93	13	-11	7	12.2720
5	-12	113	-83	106	-9	104	19.1300
6	41	65	-58	57	37	60	19.2826
7	49	202	-39	189	44	184	20.2861
8	92	-61	0	-49	82	-53	18.0941
9	95	167	0	149	82	144	14.0358
10	124	-15	51	-8	113	-12	22.3442
11	157	152	48	143	134	138	14.2818
12	170	-8	81	-1	153	-5	20.1257
13	170	-16	71	-9	154	-12	21.4648

Time = 0.3 (Frame 2)

1	-158	-26	-218	-22	-137	-21	15.9601
2	-85	-7	-161	3	-76	-3	20.0576
3	-34	-71	-115	-53	-32	-59	22.9181
4	-33	-95	-118	-86	-30	-81	19.5996
5	79	205	-11	183	71	180	16.4573
6	85	-19	-3	-8	76	-13	17.7225
7	103	-73	5	-56	92	-61	17.7855
8	125	-43	47	-30	115	-34	23.7157
9	143	56	33	60	126	54	16.6384
10	145	138	46	125	126	126	15.4135
11	172	168	59	158	150	152	15.9483
12	204	-50	96	-35	181	-40	17.9549
13	207	50	85	52	185	49	19.1903

INERTIAL NAVIGATION SENSOR INTEGRATED MOTION ANALYSIS FOR AUTONOMOUS VEHICLE NAVIGATION

Barry Roberts and Bir Bhanu
Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

ABSTRACT

Many types of existing vehicles contain an inertial navigation system (INS) which can be utilized to greatly improve the performance of motion analysis techniques and make them useful for practical military and civilian applications. This paper presents the results obtained with a maximally passive system of obstacle detection for ground-based vehicles and rotorcraft. Automatic detection of these obstacles and the necessary guidance and control actions triggered by such detection would facilitate autonomous vehicle navigation. Our approach to obstacle detection employs motion analysis of imagery collected by a passive sensor during vehicle travel to generate range measurements to world points within the field of view of the sensor. The approach makes use of INS data to improve interest point selection, matching of the interest points, and the subsequent motion detection, tracking, and obstacle detection. In this paper, we concentrate on the results obtained using lab and outdoor imagery. The range measurements that are made by INS integrated motion analysis are compared to a limited amount of ground truth that is available.

1. INTRODUCTION

A variety of active sensor-based techniques for obstacle detection have been explored to date.^{1,6,11} These approaches mainly focus on the processing of laser range (ladar) imagery and millimeter wave (MMW) radar data. In our approach, we prefer a passive sensor which will enable the vehicle to be covert and therefore minimize any possible threat to the vehicle and the pilot. Of equal importance are the field of view, resolution of the data used for obstacle detection and the access time of such data. Both MMW and ladar suffer in one of these categories.

Passive sensors, such as a TV camera, are also being used to detect obstacles for ground vehicles.^{2,3,7,8} However, state-of-the-art motion analysis techniques for obstacle detection are not robust and reliable enough for many practical applications. Many of these techniques require that unrealistic constraints be placed on the input data in order to make them work. The largest sources of error are unknown sensor motion and incomplete/ambiguous information in the sensed image data. However, many types of land and air vehicles contain an INS whose output can be used for applications beyond the original intent of the system. Within such vehicles, the INS information can be used to greatly simplify some of the functionalities normally provided by computer vision, such as obstacle detection, target motion detection, target tracking, etc.

In this paper, we briefly describe the use of INS measurements to enhance the quality and robustness of motion analysis techniques for obstacle detection and thereby provide vehicles with new functionality and capability. For a detailed algorithmic description of our approach, the reader is referred to the references.^{2,3} The objective of the work presented in this paper is to present the results of our obstacle detection approach when applied to sequences of indoor (laboratory) and outdoor imagery that has associated INS data.

In Section 2, we briefly review our approach to motion analysis by describing the fundamental details of the technique. Section 3 describes the results we have obtained with our INS integrated motion analysis approach. Finally, Section 4 provides the conclusions of the paper.

2. INERTIAL SENSOR INTEGRATED MOTION ANALYSIS

The purpose of this section is to describe the inertial sensor integrated motion analysis approach we have undertaken. The block diagram of this system is illustrated in Figure 1. The system uses inertial sensor integrated motion analysis, scene analysis, and selective applications of active sensors to provide an obstacle detection capability.²

As shown in Figure 2, the data input to the obstacle detection algorithm consists of a sequence of digitized video or FLIR frames that are accompanied by inertial data consisting of rotational and translational velocities. This information, coupled with the temporal sampling interval between frames, is used to compute the distance vector, \vec{d} , between each pair of frames and the roll, pitch and yaw angles, (ϕ, θ, ψ) , of each frame. Both \vec{d} and (ϕ, θ, ψ) are crucial to the success of the algorithm, as will be described later.

The blocks shown in Figure 2 define the major steps involved within the ODIN (Obstacle Detection using Inertial Navigation data) motion analysis algorithm suite. In the subsections that follow, we briefly address the function of these boxes.

2.1 DISTINGUISHED FEATURES

The features within the imagery (TV or FLIR) that are most prominent and distinguished mark the world points to which range measurements will be made. These prominent world points, known as *interest points*, are (by definition) those points which have the highest promise of repeated extraction throughout multiple frames. The interest points within the field-of-view of the monocular sensor are of fundamental and critical importance to motion analysis calculations. In the following subsections, the extraction and subsequent use of interest points is briefly described.

2.1.1 Image Segmentation -- Unfortunately, not all regions within a scene can contain reliable interest points. We employ scene analysis techniques to ascertain the *goodness* of regions prior to interest point selection.⁴ Hence, the interest point extraction routine takes as input a segmentation of the original image and returns n_j , $0 \leq j \leq N$, interest points in each of the N segments. The value of n_j for segment j is proportional to the segment size and other segment features. More than n_j interest points can exist per segment; only the points with the highest *interestingness* values are reported. The result of incorporating scene segmentation results into interest point extraction is that for a given scene, the interest points are more uniformly distributed.

2.1.2 Interest Point Selection -- We compute a set of distinguishable points by passing an operator, which is a combination of the Hessian and Laplacian operators,⁹ over each frame of imagery. The operator, I , takes the form

$$I(g) = g_{xx}g_{yy} - g_{xy}^2 - k(g_{xx} + g_{yy})^2$$

where k is a constant and g is the local gray level function and g_{xx} for example, is the local 2nd derivative in the x direction. At the current time, local maxima of I are selected as interest points.

2.1.3 Interest Point Derotation -- To aid the process of interest point matching, we must make it seem as though image plane $M+1$ is parallel to image plane M . If this is done, the FOE and pairs of interest points in frames M and $M+1$ that match would ideally be co-linear should the image planes be superimposed.

The pixels in the image plane can be described in the sensor's 3-D coordinate frame by the vector (F, y, z) , where F is the focal length of the sensor. To make the image planes parallel, derotation is performed for each vector, (F, y_i, z_i) that corresponds to each interest point in frame B . The equation for the derotation transformation and projection (in homogeneous coordinates) is

$$\begin{bmatrix} F \\ y_i' \\ z_i' \\ 1 \end{bmatrix} = P R_{\phi_A}^{-1} R_{\theta_A}^{-1} R_{\psi_A}^{-1} R_{\psi_B} R_{\theta_B} R_{\phi_B} \begin{bmatrix} F \\ y_i \\ z_i \\ 1 \end{bmatrix} = P C_{NED}^A C_P^{NED} \begin{bmatrix} F \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

where NED (*north, east, down*) is the coordinate frame in which inertial measurements are made.

The matrix P projects a world point onto an image plane and is used to compute the FOE, $FOE = P \vec{d}$, where $\vec{d} = \nabla \Delta t$. The matrix C_{NED}^A converts points described in the NED coordinate frame into an equivalent description within a coordinate frame parallel to the A coordinate frame. Likewise, the matrix C_P^{NED} converts the descriptions of points in the B coordinate frame into descriptions in a coordinate frame parallel to NED.

2.1.4 Interest Point Matching -- The goal of interest point matching is to identify and store the best match in frame M for each interest point in frame N ($= M+1$), (F, y_{Nj}, z_{Nj}) . Several metrics/constraints assist us in this task. To determine the candidate matches to (F, y_{Nj}, z_{Nj}) , each of the interest points in frame M is examined with the successive use of four metrics. The *first metric* requires that the interestingness, edge magnitude, and edge direction of both points of a candidate match are nearly equivalent. Edge direction is treated differently than the other parameters. We recognize that when an edge's normal is perpendicular to the line connecting edge pixels to the FOE, the interest points on this edge will not be reliably matched and ranged. This is due to the way that interest points travel radially away from

the FOE. The interest points along these special edges are weighted differently because they are more difficult to track and therefore less reliable.

The *second metric* makes certain that candidate matches lie within a cone shaped region, with apex at the FOE, bisected by the line joining the FOE and the interest point in frame M . The *third metric* restricts all candidate matches in frame M to lie closer to the FOE than the points in frame N (as physical laws would predict for stationary objects). The *fourth metric* constrains the distance between an interest point and its candidate matches. This is done by imposing maximum and minimum range constraints upon the resulting match.

Hence, the second, third and fourth metrics combine such that for an interest point in frame M , M_j , to be a candidate match to point N_j , N_j must lie in a region which is shaped like a sector of an annulus.

The reasoning behind the maximum and minimum range restriction is that world objects of range less than R_{\min} are not possible considering the sensor mounting location on the vehicle and its field of regard. In another way, world objects that would lie closer than some R_{\min} have been visible for some time and have been detected and therefore avoided by the vehicle navigator (machine or human). Likewise, objects at a range greater than R_{\max} are not yet of concern to the vehicle.

2.1.5 Matching and Range Confidence Factors -- We further improve range computations (based upon three or more sequential frames) by predicting and smoothing the range to each interest point that can be tracked through multiple frames. The procedure for prediction and smoothing of range using multiple frames is to compute, for all interest points in a pair of images, the matching confidence, confidence in range, and predicted ranges. Once the confidences and predicted range are computed, thresholds are applied and a smoothed range is computed.

The *Matching Confidence* of the i th point in frame A is given by

$$C_{Mi}^A = w_1 \left[1 - \frac{|I_{Ai} - I_{Bi}|}{\max I_{AB} - \min I_{AB}} \right] + w_2 \left[1 - \frac{|d_i^n - \min d_i^n|}{\max d_i^n - \min d_i^n} \right] + w_3 |\hat{\phi} \cdot \hat{a}|$$

where

$$\max I_{AB} = \max_i (I_{Ai}, I_{Bi}), \quad \min I_{AB} = \min_i (I_{Ai}, I_{Bi}),$$

$$w_1, w_2, w_3 \geq 0 \quad \text{and} \quad w_1 + w_2 + w_3 = 1$$

The variable I_{Xi} is the *interestingness* of the i th point in frame X and d_i is the projection of the i th point onto the line connecting its match point with the FOE. The unit vector $\hat{\phi}$ is in the direction of the line connecting the FOE and the i th point in frame A . The unit vector \hat{a} represents the normal to the edge on which the i th interest point is located. The purpose of $|\hat{\phi} \cdot \hat{a}|$ is to cause the match confidence to fall when $\hat{\phi}$ and \hat{a} are perpendicular (see section 2.1.4).

The *Range Confidence*, C_{Ri}^X , of the i th point in frame X is given by the following set of equations:

$$R_{i \text{ final}}^0 = R_{i \text{ predicted}}^0 = R_{i \text{ measured}}^0 \quad \text{and} \quad C_{Ri}^0 = 1 \quad (1)$$

$$R_i^n \text{ predicted} = R_i^{n-1} \text{ final} - \text{velocity}_i \times \text{time} \quad (2)$$

If $(R_i^n \text{ predicted} \leq 0)$ then

$$R_i^n \text{ final} = R_i^n \text{ predicted} = R_i^n \text{ measured} \text{ and } C_{Ri}^n = 1 \quad (3)$$

Else, if $(\alpha < \frac{R_i^n \text{ predicted}}{R_i^n \text{ measured}} < 2 - \alpha)$ then

$$C_{Ri}^n = C_{Mi}^n \left[1 - \frac{|R_i^n \text{ measured} - R_i^n \text{ predicted}|}{R_i^n \text{ measured} + R_i^n \text{ predicted}} \right] \quad (4)$$

$$R_i^n \text{ final} = R_i^n \text{ measured} + (1 - C_{Ri}^n) D, \quad (5)$$

$$\text{where, } D = [R_i^n \text{ predicted} - R_i^n \text{ measured}] \quad (6)$$

If $(R_i^n \text{ final} < 0)$ then

$$R_i^n \text{ final} = \frac{R_i^n \text{ measured} R_i^n \text{ predicted}}{R_i^n \text{ measured} + R_i^n \text{ predicted}} \quad (7)$$

The variable α is a user defined parameter that controls the range of the ratio $R_i^n \text{ predicted} / R_i^n \text{ measured}$.

2.2 RANGE CALCULATION

Given the result of interest point matching, range can be computed to each match. Given these sparse range measurements, a range or obstacle map can be constructed. The obstacle map can take many forms,^{5,10} the simplest of which consists of a display of bearing versus range. In what follows, range calculation is described and the important issue of range interpolation is discussed.

Given pairs of interest point matches between two successive image frames and the translational velocity between frames, it becomes possible to compute the range to the object on which the interest points lie. Our approach to range computation is described by the equation

$$R = \Delta Z \frac{x' - x_f}{x' - x} \frac{1}{\cos \alpha_A}$$

where

x_f = the distance between the FOE and the center of the image plane,

x = the distance between the pixel in frame A and the center of the image plane,

x' = the distance between the pixel in frame B and the center of the image plane,

$\Delta Z = |\mathbf{v}| \Delta t \cos \alpha_F$ = the distance traversed in one frame time, Δt , as measured along the axis of the line of sight,

α_F = the angle between the velocity vector and the line of sight,

α_A = the angle between the vector pointing to the world object and the line of sight,

$x' - x_f$ = the distance in the image plane between (F, y_{B_j}, z_{B_j}) and the FOE, and

$x' - x$ = the distance in the image plane between (F, y_{B_j}, z_{B_j}) and (F, y_{A_i}, z_{A_i}) .

These variables are illustrated in Figure 3. The range equation is used to compute the distance to a world point relative to the lens center of frame A (similar equations would compute the distance from the lens center of frame B). The accuracy of the range measurements that result is very sensitive to the accuracy of the interest extraction process, the matching process, and the accuracy of the INS data.

3. EXPERIMENTAL RESULTS

Our inertial navigation sensor integrated motion analysis algorithm has been used to generate range samples on both indoor (laboratory) imagery (plus sensor motion parameters measured without an INS) and outdoor imagery, with real INS data, obtained from onboard a moving vehicle. In this section, we describe the conditions under which the data was created/collected and provide images illustrating the results of the major steps in the motion analysis algorithm.

3.1 INDOOR DATA

A sequence of imagery was collected inside of a computer lab by moving a camera forward in discrete 2.0 ft steps. The velocity and attitude of the camera were estimated as 2 ft/sec forward with no attitude changes throughout all 5 frames. Five frames of the resulting imagery are displayed in Figure 4. The field of view of the camera used to collect these images is $43.2^\circ \times 18^\circ$ and the focal length = 12.5 mm. An example of the processing that was performed is displayed in Figure 5. The results of the various steps are illustrated; (a) segmentation, (b-c) interest point extraction and derotation, (d) matching, and (e) computed range. Note that only the interest points in the second frame of the pair are derotated. The derotated locations of the points are represented by diamonds and their original positions are shown as squares. The points in the first image of the pair are denoted by circles.

The image in Figure 6 is the cumulative result of processing the 5 frame sequence shown in Figure 4. Ideally, we would see a chain of connected circles which would denote the location of strong interest points which were tracked through all 5 frames. In this case, we see very few instances of chains of circles due in part to the large separation between frames.

For the lab images in Figure 4, we have a limited amount of *ground truth* information. By ground truth, we mean that we have actually measured the range between the camera and various lab objects. With this information, we can study the accuracy of the motion analysis generated range values. Figure 7 shows the locations of the objects for which ground truth exists. Table 1 provides a comparison of ground truth range values and the range values generated through motion analysis for 4 pairs of imagery. Note that some motion analysis range values are missing. This is because a computer selected interest point did not fall on the corresponding ground truthed object. For the table entries that are provided, there existed an interest point that fell on the corresponding ground truthed object.

Table 2 illustrates the effect that the smoothing filter has on interest point range values when a world object is tracked through multiple frames. The *final* range values, as described in equations 1-6, are dependent on the listed values of match confidence, range confidence, predicted range, and measured range.

To explore the sensitivity of motion analysis range calculations to INS errors, we have added noise to the INS data

and performed the motion analysis. Two different amounts of INS sensor drift were added to the original INS measurements. Tables 3 and 4 show the velocity and attitude of the sensor at the time of each frame acquisition. Note that the terms in the velocity columns are actually the amount of error that is added to the actual velocity. The velocity error, composed mainly of positive azimuthal drift, was significant enough to move the FOE anywhere from 8 to 15 pixels to the right. The first case allows less drift to occur as compared to the second case. The locations of the interest points did not change in either case.

Figure 8 shows the results of matching the two frames in Figure 5(b) and 5(c) when both cases of INS error are added. The difference between the two cases is minimal this early in the sequence. The FOE location differs only by 4 pixels between the frames in Figures 8(a) and 8(b). The number of matches didn't change between Figures 8(a) and 8(b) because of the width of the sector of an annulus which constrains the matching process. Of course, the range resulting from these matches will fluctuate noticeably.

In Tables 5 and 6, we show a comparison of ground truth range values and the range values generated through motion analysis for 4 pairs of imagery with noisy INS data of the first and second cases respectively.

3.2 OUTDOOR DATA

A sequence of outdoor imagery was collected along with INS data generated by a Honeywell HG1050. Table 7 indicates the roll, pitch, yaw, and velocity of the camera associated with the sequence of outdoor frames that were used. The velocity and attitude measurements are made in the coordinate frame of the INS. Figure 9 illustrates the hardware used to collect the imagery. The video frames were stored on optical disk at a 5 Hz rate that was synchronized with the collection of INS data. The INS data was collected at a 50 Hz rate and stored with a time stamp. To determine the correspondence between video frame and INS data packet, one has only to read the time stamp written on the frame when stored and find the corresponding INS data packet.

A five frame sequence of the collected imagery is presented in Figure 10. The field of view of the camera used to collect these images is $32.6^\circ \times 22.1^\circ$ and the focal length = 15.1 mm. The elapsed time between each pair of frames for this experiment was 0.3 seconds. An example of the processing that was performed is displayed in Figure 11. The results of the various steps are illustrated; (a) segmentation, (b-c) interest point extraction and derotation, (d) matching, and (e) computed range. The image in Figure 12 is the cumulative result of processing the 5 frame sequence shown in Figure 10.

As in the case of indoor imagery, we again have a limited amount of ground truth information. Figure 13 shows the locations of the objects for which ground truth exists. Table 8 provides a comparison of ground truth range values and the range values generated through motion analysis for 4 pairs of imagery. Again, note that some motion analysis range values are missing because no interest points fell on the appropriate ground truthed object.

4. CONCLUSIONS

We have presented our latest work on INS integrated motion analysis. The most important lesson learned from this research is that the incorporation of INS data into the

motion analysis problem greatly improves the analysis and makes the process more robust. We have learned the benefit of scene analysis which can be used to guide interest point extraction and surface interpolation, and we have gained insight into the sensitivity of the motion analysis ranging to interest point position shifts and INS errors.

Our ongoing efforts have the ultimate goal of developing a complete, fieldable system for obstacle detection during rotorcraft low altitude flight.

ACKNOWLEDGEMENTS

This material is based upon work supported by NASA under contract NAS2-12800 and DARPA/U.S. Army ETL contract DACA 76-86-C-0017. The authors would like to thank Dave Duncan, J.C. Ming, and Scott Snyder for their input to this work.

REFERENCES

1. "Millimeter-Wave Radar May Enhance Safety of Helicopter Flights," *Aviation Week and Space Technology*, p. 103 (July 1987).
2. B. Bhanu and B. Roberts, "Obstacle Detection During Rotorcraft Low-Altitude Flight," Annual Technical Report for NASA-Ames (April, 1989).
3. B. Bhanu, B. Roberts, and J.C. Ming, "Inertial Navigation Sensor Integrated Motion Analysis," Proceedings of DARPA Image Understanding Workshop, pp. 747-763 (May, 1989).
4. B. Bhanu and P. Symosek, "Interpretation of Terrain Using Hierarchical Symbolic Grouping for Multi-Spectral Images," *Proc. DARPA Image Understanding Workshop*, pp. 466-474 (February 1987).
5. V.H.L. Cheng, "Obstacle-Avoidance Automatic Guidance - A Concept-Development Study," Proceedings of AIAA Guidance, Navigation and Control Conference, pp. 1142-1152 (August 1988).
6. M.J. Daily, J.G. Harris, and K. Reiser, "Detecting Obstacles in Range Imagery," Proceedings of DARPA Image Understanding Workshop, pp. 87-97 (February 1987).
7. R. Dutta, R. Manmatha, E.M. Riseman, and M.A. Snyder, "Issues in Extracting Motion Parameters and Depth from Approximate Translational Motion," Proceedings of DARPA Image Understanding Workshop, pp. 945-960 (April, 1988).
8. L. Matthies, R. Szeliski, and T. Kanade, "Kalman Filter-Based Algorithms for Estimating Depth from Image Sequences," Proceedings of DARPA Image Understanding Workshop, pp. 199-213 (April, 1988).
9. H.H. Nagel, "Displacement Vectors Derived from Second-Order Intensity Variations in Image Sequences," *Computer Vision, Graphics, and Image Processing* 21 pp. 85-117 (1983).
10. F.W. Smith and M. Streicker, "Passive Ranging from a Moving Vehicle via Optical Flow Measurement," *SPIE: Applications of Digital Image Processing* 829 pp. 310-317 (1987).
11. C. Thorpe, S. Schafer, and T. Kanade, "Vision and Navigation for the Carnegie Mellon Navlab," Proceedings of DARPA Image Understanding Workshop, pp. 143-153 (February 1987).

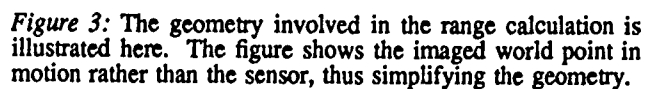
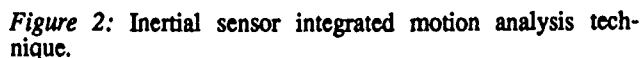
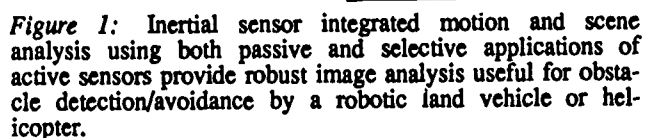




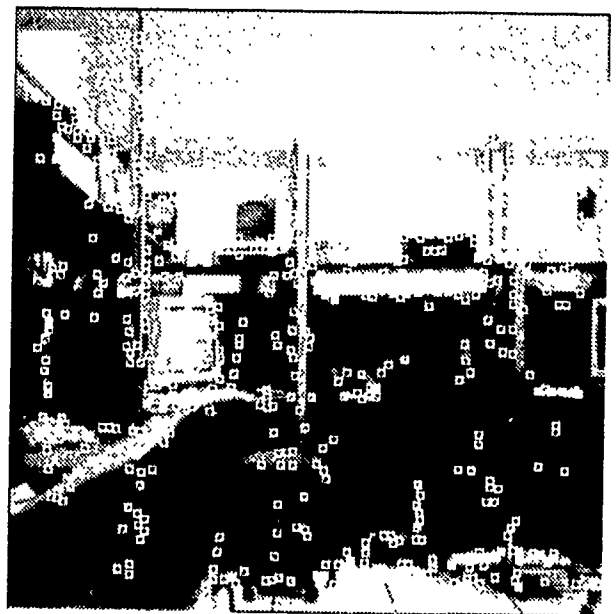
Figure 4: The five frame sequence of indoor (lab) imagery.



(a)

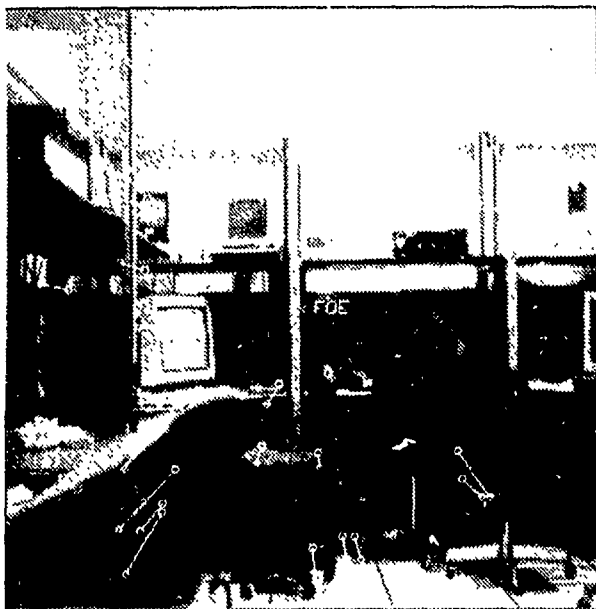


(b)



(c)

Figure 5: The results of processing one pair of the indoor imagery: (a) the segmentation of both frames, (b) the interest points in the 1st frame, (c) the interest points in the 2nd frame



(d)



(e)

Figure 6: The cumulative result of processing five frames of indoor imagery. Every interest point which was matched and assigned a range is superimposed here on the first frame of the sequence.



Figure 5: The results of processing one pair of the indoor imagery: (d) the set of matched points, (e) the range to the matched points.

Figure 7: The locations of the lab points which had associated ground truth information.



(a)



(b)

Figure 8: The results of the matching of the frames in Figures 4(a) and 4(b) when noise was added to the INS information: (a) case one, (b) case two.

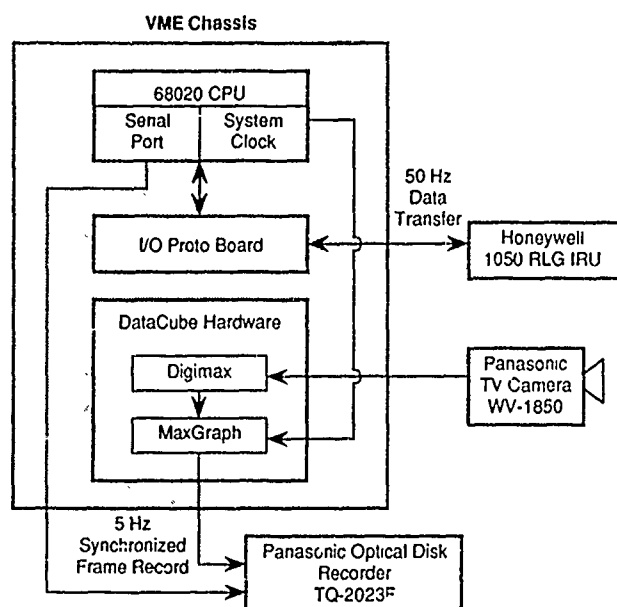
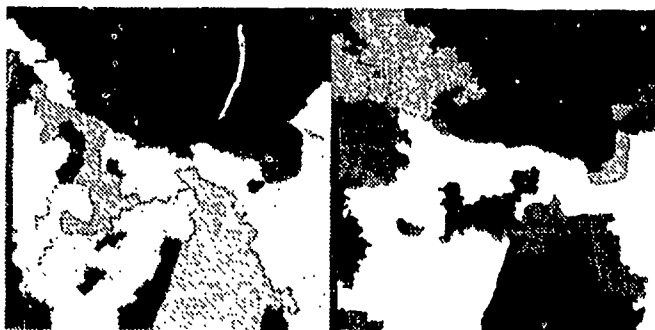


Figure 9: The hardware used to collect the outdoor imagery and INS data.



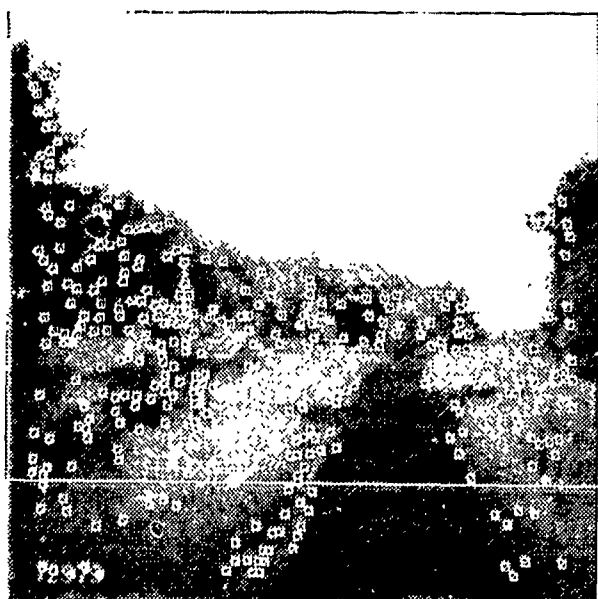
Figure 10: The five frame sequence of outdoor imagery.



(a)



(b)



(c)



(d)



(e)

Figure 11: The results of processing one pair of the outdoor imagery: (a) the segmentation of both frames, (b) the interest points in the 1st frame, (c) the interest points in the 2nd frame, (d) the set of matched points, (e) the range to the matched points.



Figure 12: The cumulative result of processing five frames of outdoor imagery. Every interest point which was matched and assigned a range is superimposed here on the first frame of the sequence.



Figure 13: The locations of the world points which had associated ground truth information.

Table 1: A comparison of ground truth and motion analysis range values for the indoor imagery. The columns labeled *Actual* contain the ground truth values and the columns labeled *ODIN* contain the motion analysis generated range.

Ground Truth Location	A-B Range (ft)		B-C Range (ft)		C-D Range (ft)		D-E Range (ft)	
	Actual	ODIN	Actual	ODIN	Actual	ODIN	Actual	ODIN
A.	13.76	--	11.80	9.45	9.84	7.17	7.57	10.90
B.	14.28	15.42	12.33	--	10.40	11.77	8.50	--
C.	14.00	14.72	12.05	--	10.12	9.52	8.22	--
D.	20.95	--	19.00	16.99	17.02	19.72	15.07	13.77
E.	18.13	--	16.16	21.41	14.20	--	12.24	11.34
F.	20.64	--	18.65	17.88	16.67	--	14.70	12.03
G.	21.14	--	19.14	19.36	17.15	23.52	15.16	12.64
H.	20.14	18.50	18.14	15.31	16.14	12.37	14.14	11.92
I.	22.37	--	20.37	--	18.38	--	16.38	--
J.	23.00	21.80	21.02	20.96	19.0	--	17.08	--
K.	20.66	21.60	18.73	18.30	16.80	15.28	14.91	--

Table 2: A set of 4 example interest points which were extracted from 3 consecutive frames. One can see the effect of the smoothing filter in generating the final range value and the filter's dependence upon the confidence factors.

Range smoothing results				
Range (ft)	A-B	B-C	C-D	D-E
Measured	--	12.78	12.27	--
Predicted	--	--	10.88	--
Final	--	12.78	12.02	--
Match Confidence		0.61	0.88	--
Range Confidence	--	1.0	0.83	--
Measured	21.79	17.70	--	--
Predicted	--	19.81	--	--
Final	21.79	18.32	--	--
Match Confidence	0.88	0.75	--	--
Range Confidence	1.0	0.71	--	--

Measured	--	--	22.33	15.89
Predicted	--	--	--	20.39
Final	--	--	22.33	17.55
Match Confidence	--	--	0.57	0.72
Range Confidence	--	--	1.0	0.63
Measured	--	11.04	18.88	--
Predicted	--	--	9.07	--
Final	--	11.04	12.72	--
Match Confidence	--	0.50	0.57	--
Range Confidence	--	1.0	0.37	--

Table 3: The first set attitude and velocity error terms which were used to add noise to the INS data of the indoor imagery.

Frame	Attitude (radians)			Velocity (ft/s)		
	roll	pitch	yaw	v_{north}	v_{east}	v_{down}
A	1.335e-04	-6.938e-05	-1.594e-02	0.0	0.0	0.0
B	1.371e-04	-6.795e-05	-1.594e-02	30.51e-06	0.68e-03	0.0
C	1.369e-04	-6.778e-05	-1.594e-02	63.32e-06	1.36e-03	0.0
D	1.374e-04	-6.802e-05	-1.594e-02	92.85e-06	2.03e-03	0.0
E	1.375e-04	-6.817e-05	-1.594e-02	127.30e-06	2.71e-03	0.0

Table 4: The second set attitude and velocity error terms which were used to add noise to the INS data of the indoor imagery.

Frame	Attitude (radians)			Velocity (ft/s)		
	roll	pitch	yaw	v_{north}	v_{east}	v_{down}
A	1.36e-04	-6.75e-05	-1.5942e-02	0.31e-03	6.7e-03	0.0
B	1.35e-04	-6.36e-05	-1.5940e-02	0.48e-03	13.4e-03	0.0
C	1.34e-04	-6.06e-05	-1.5947e-02	0.53e-03	20.0e-03	0.0
D	1.36e-04	-6.54e-05	-1.5944e-02	0.70e-03	26.5e-03	0.0
E	1.41e-04	-6.33e-05	-1.5944e-02	0.95e-03	33.2e-03	0.0

Table 5: A comparison of ground truth and motion analysis range values. This table represents the results obtained when the first set of INS errors were added.

Ground Truth Location	A-B Range (ft)		B-C Range (ft)		C-D Range (ft)		D-E Range (ft)	
	Actual	ODIN	Actual	ODIN	Actual	ODIN	Actual	ODIN
A.	13.76	--	11.80	--	9.84	--	7.57	--
B.	14.28	--	12.33	13.71	10.40	--	8.50	--
C.	14.00	15.71	12.05	13.84	10.12	9.55	8.22	--
D.	20.95	--	19.00	--	17.02	23.49	15.07	14.54
E.	18.13	--	16.16	--	14.20	--	12.24	--
F.	20.64	--	18.65	--	16.67	--	14.70	14.25
G.	21.14	--	19.14	19.99	17.15	22.94	15.16	13.81
H.	20.14	--	18.14	15.90	16.14	--	14.14	14.67
I.	22.37	--	20.37	23.66	18.38	--	16.38	--
J.	23.00	20.14	21.02	19.30	19.04	--	17.08	--
K.	20.66	20.54	18.73	17.42	16.8	14.18	14.91	--

Table 6: A comparison of ground truth and motion analysis range values. This table represents the results obtained when the second set of INS errors were added.

Ground Truth Location	A-B Range (ft)		B-C Range (ft)		C-D Range (ft)		D-E Range (ft)	
	Actual	ODIN	Actual	ODIN	Actual	ODIN	Actual	ODIN
A.	13.76	--	11.80	10.12	9.84	--	7.57	--
B.	14.28	16.91	12.33	--	10.40	--	8.50	--
C.	14.00	16.01	12.05	14.21	10.12	9.84	8.22	--
D.	20.95	--	19.00	--	17.02	23.82	15.07	15.25
E.	18.13	--	16.16	--	14.20	--	12.24	--
F.	20.64	--	18.65	--	16.67	--	14.70	14.05
G.	21.14	--	19.14	18.65	17.15	--	15.16	14.77
H.	20.14	--	18.14	--	16.14	--	14.14	12.37
I.	22.37	--	20.37	--	18.38	--	16.38	15.38
J.	23.00	--	21.02	18.07	19.04	--	17.08	--
K.	20.66	20.20	18.73	16.99	16.8	13.71	14.91	--

Table 7: The actual attitude and velocity measurements made simultaneously with the acquisition of the outdoor imagery. These measurements are in the coordinate frame of the INS. The vehicle was moving roughly E-NE.

Frame	Attitude (radians)			Velocity (ft/s)		
	roll	pitch	yaw	v_{north}	v_{east}	v_{down}
A	3.49e-02	2.72e-02	1.33	2.24	8.36	-0.149
B	2.99e-02	2.75e-02	1.328	2.30	8.32	-0.150
C	2.61e-02	2.90e-02	1.327	2.23	8.23	-0.150
D	2.42e-02	3.01e-02	1.326	2.19	8.23	-0.120
E	2.53e-02	2.99e-02	1.325	2.01	8.23	-0.133

Table 8: A comparison of ground truth and motion analysis range values for the outdoor imagery. The columns labeled *Actual* contain the ground truth values and the columns labeled *ODIN* contain the motion analysis generated range.

Ground Truth Location	A-B Range (ft)		B-C Range (ft)		C-D Range (ft)		D-E Range (ft)	
	Actual	ODIN	Actual	ODIN	Actual	ODIN	Actual	ODIN
A. 1st Telephone pole	231	185.	228	276.	226	245	223	202
B. 2nd Telephone pole	486	367	483	366	480	427	478	--
C. Treeline #1	502	--	499	--	497	430	494	--
D. Treeline #2	665	300	663	298	660	--	658	446
E. Treeline #4	388	--	385	--	383	255	380	--
F. Pole by gate	214	298	212	--	209	236	206	175
G. Red light post (closest to road)	153	186	150	322	148	--	145	165
H. Red light post (closest to gate)	156	167	153	343	151	114	148	157
I. Fence Post by gate (West end, closest)	169	160	167	172	164	--	162	161
J. Fence Post by gate (West end, farthest)	156	209	153	--	151	165	148	153

Temporal Integration of Visual Surface Reconstruction

Joachim Heel*

545 Technology Square, Room # 826
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139.

Satyajit Rao

545 Technology Square, Room # 809
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139.

Abstract

This paper presents a solution for the problem of obtaining and improving an estimate of the 3-D structure of a scene from a sequence of images. It establishes a connection between surface reconstruction techniques used in single-frame analysis and Kalman filtering which is used for multi-frame integration of information. The formulation allows for multiple, independent sources of structure information to be combined over time. Based on this concept, algorithms are presented that integrate structure from motion, stereo and shading information over time and improve significantly over single frame estimates on a variety of image examples, some of which are presented towards the end of the paper.

1 Introduction

Recovering the 3-D structure of a scene from its images is an eminent problem in computer vision. It has been studied extensively for shape from stereo, shading, motion and non-vision sources of distance data. Most work has focused on image snapshots at a fixed instant in time with the exception of work in motion vision in which information is inherently available as sequences of images.

The work in "instantaneous" structure estimation has focused on recovering smooth surfaces with discontinuities that are compatible with the image data. Examples of visual surface reconstruction are Grimson's stereo algorithm [7] and Horn's shape from shading algorithm [10]. Based on theoretical foundations by Geman and Geman [6] and Marroquin [14], the study of surface reconstruction evolved detached from vision for some time, until Terzopoulos [19], Blake and Zisserman [1], Poggio, Gamble and Little [17] and Geiger and Girosi [4] explored applications of the theory to vision using both stochastic relaxation algorithms which are computationally quite intensive and more efficient deterministic approximations.

On the other hand, researchers in motion vision have been considering, how depth information may be ob-

tained from a sequence of images. The objective was to improve the rather poor structure estimates that result from using only two temporally sequential frames. One of the investigated alternatives was the epipolar image technique of Bolles and Baker in which images are "stacked" temporally and slices through this stack are analyzed (Bolles and Baker [2], Yamamoto [21]). A more widely studied approach is based on recursive estimation or Kalman filtering. Ullman's incremental rigidity scheme [20] pointed in this direction and Brodia and Chellappa [3] presented a first application. With some significant modifications this finally lead to practical algorithms for real images as shown by Matthies, Szeliski and Kanade [15] and Heel [8], [9].

In the light of these developments a rather obvious question is how both approaches could be combined. Is it possible to integrate information from an arbitrary source of depth data over time to reduce the effects of noise by coupling the surface reconstruction techniques with the recursive estimation procedure? Is it further possible to combine depth information from multiple sources in this way?

In this paper we show a fundamental and very simple connection between the surface reconstruction techniques widely used in vision and the recursive estimation procedures that have recently been developed.

2 Surface reconstruction and recursive estimation theory

In this section we will briefly review basic concepts in surface reconstruction and recursive estimation theory. The presentation is extremely simplified and focused on the aspects of both fields that are relevant to our specific problem of depth estimation from image data.

2.1 Surface reconstruction

A surface estimated from image data can be represented by giving the distance Z along the optical axis between the focal point of the imaging system and the surface for each pixel location in the image as shown in figure 1. If this is done for each pixel location (i, j) a depth map Z_{ij} is obtained.

Using this concept we will now consider the following problem formulation (illustrated in figure 1): We are given two depth map estimates Z_1 and Z_2 at pixel locations (i, j) of a discrete grid G of the same surface.

*This research was conducted at the MIT AI Lab with support from ARPA contract DACA 76-85-K-0685 and ONR contract N00014-85-K-0124.

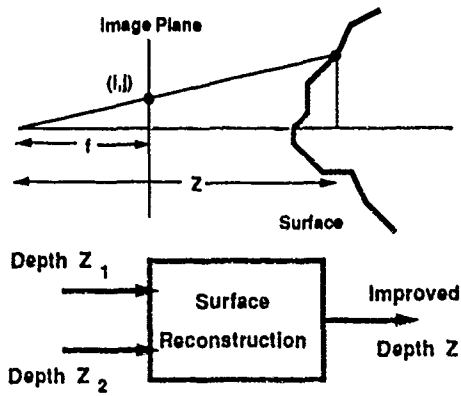


Figure 1: Top: The measurement of "depth" for a given pixel location. Bottom: Surface reconstruction problem depiction.

Our objective is to compute a depth map Z that is as close as possible to both Z_1 and Z_2 as a "better" estimate of the surface structure. This problem is formulated as an optimization problem:

$$\min_Z \sum_{i,j \in G} \lambda_1 (Z_{ij} - Z_{1ij})^2 + \lambda_2 (Z_{ij} - Z_{2ij})^2 \quad (1)$$

where the parameters λ_1 and λ_2 can be used to weight either input depth more strongly. The optimal Z that minimizes the sum of squared errors to both Z_1 and Z_2 is

$$Z_{ij} = \frac{\lambda_1 Z_{1ij} + \lambda_2 Z_{2ij}}{\lambda_1 + \lambda_2} \quad (2)$$

The smoothness constraint can be imposed on the reconstructed surface by adding a term that penalizes large gradients to the optimization criterion

$$\min_Z \sum_{i,j \in G} \lambda_1 (Z_{ij} - Z_{1ij})^2 + \lambda_2 (Z_{ij} - Z_{2ij})^2 + \mu (\nabla Z_{ij})^2 \quad (3)$$

where ∇Z_{ij} denotes a discrete approximation of the magnitude gradient of Z at location (i, j) . In the continuous domain, the solution can be obtained very elegantly using the calculus of variation (see Horn [10]). In the discrete version a careful consideration of the indices involved leads to the same result

$$\lambda_1 (Z_{ij} - Z_{1ij}) + \lambda_2 (Z_{ij} - Z_{2ij}) - \mu \Delta Z_{ij} = 0 \quad (4)$$

where ΔZ_{ij} is a discrete approximation to the Laplacian of Z at location (i, j) . With a suitable discrete approximation of the Laplacian, an iterative solution scheme (see Horn [10]) can be developed for the discrete difference equation (4)

$$Z_{ij}^{(n+1)} = \frac{\lambda_1 Z_{1ij} + \lambda_2 Z_{2ij} + \mu' \bar{Z}_{ij}^{(n)}}{\lambda_1 + \lambda_2 + \mu'} \quad (5)$$

where n indicates the iteration number, \bar{Z}_{ij} is a local average of Z at location (i, j) and μ' is proportional to μ . In addition to combining the input data, this scheme will

also allow information from neighboring pixels to propagate and thereby achieve the smoothness of the resulting surface. Additional constraints such as allowing for discontinuities (Poggio, Gamble and Little [17], Blake and Zisserman [1], Geiger and Girosi [4]) have been incorporated recently. The formalism presented here extends to this case as well although the derivation is omitted here as it is rather lengthy

2.2 Recursive estimation theory

There are many in-depth treatments of recursive estimation theory such as Gelb [5]. The presentation here is again extremely simplified and uses notation from the vision domain which is chosen to suggest the relationship to the surface reconstruction ideas introduced above.

Suppose our task is to estimate a (scalar) quantity $Z(k)$ from a sequence of measurements Z_k taken at discrete points k in time. Suppose further that the Z_k are generated by a stochastic process

$$Z_k = Z(k) + n_k \quad (6)$$

where n_k is zero mean Gaussian noise of known variance p_k . Finally, we know that the quantity Z changes over time according to the difference equation

$$Z_{k+1} = f(Z_k) \quad (7)$$

The goal is to compute at every time instance k an estimate \hat{Z}_k which is as close as possible to the true value $Z(k)$. Kalman formulated and solved this problem in a far more general case (see the literature cited above) and the resulting algorithm is referred to as a *Kalman filter*.

The Kalman filter maintains an estimate \hat{Z}_k and its variance \hat{p}_k . When a new measurement Z_k with variance p_k becomes available we perform the following *update* operation (denoted by the superscript changing from '-' to '+'):

$$\hat{Z}_k^+ = \frac{Z_k/p_k + \hat{Z}_k^-/\hat{p}_k^-}{1/p_k + 1/\hat{p}_k^-} \quad (8)$$

$$\hat{p}_k^+ = \frac{1}{1/p_k + 1/\hat{p}_k^-} \quad (9)$$

In words, the new estimate of \hat{Z}^+ is a weighted sum of the old estimate \hat{Z}^- and the new measurement Z with the inverse variances as weights. Since the depth Z changes according to the known dynamics (7) we can use this known temporal behavior to *predict* how the new improved model from the update stage would appear in the next iteration $k+1$

$$\hat{Z}_{k+1}^- = f(\hat{Z}_k^+) \quad (10)$$

$$\hat{p}_{k+1}^- = \left(\frac{\partial f}{\partial Z} \right)^2 \hat{p}_k^+ \quad (11)$$

The Kalman filter consists of recursively invoking update and predict stages on each new measurement obtained and is known to be both optimal and convergent in the case where the dynamics f are linear. We can visualize the operation of the Kalman filter with the help of a block diagram as shown in the top diagram of figure 2.

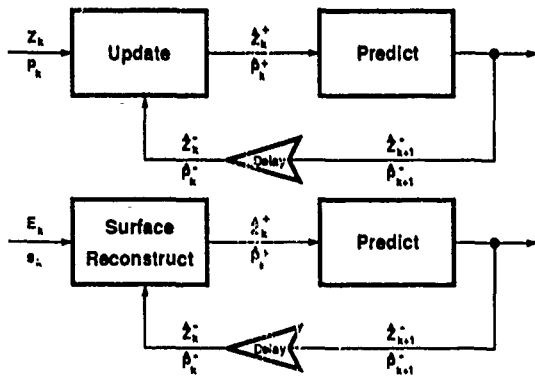


Figure 2: Top: Block diagram of a Kalman filter. Bottom: Block diagram of the temporal surface reconstruction algorithm.

The Kalman filter provides an intuitive and elegant way for the integration of depth measurement data over time. Notice, however, that the estimation process operates independently for each depth value so that the estimation of an entire depth map requires a Kalman filter for each pixel and that smoothness or discontinuity constraints on the surface cannot be incorporated.

2.3 The missing link: temporal surface reconstruction

In this section we will describe a fundamental link between the surface reconstruction techniques and the recursive estimation procedure. The ability to couple both theories will enable us to integrate structure data from independent sources in both space and time.

We will investigate the relationship between the update stage of the Kalman filter the simple surface reconstruction problem (1). In both cases, the objectives are the same: compute a depth map that best matches two input depth maps. In the Kalman filter case, the two input depth maps are the measurement Z_k and the current estimate \hat{Z}_k^- .

To understand the relationship between both cases let us suppose the parameters λ_1 and λ_2 are not constant but may vary spatially. At each pixel location (i, j) we set

$$\lambda_{1ij} = 1/p_{1ij} \quad \text{and} \quad \lambda_{2ij} = 1/p_{2ij} \quad (12)$$

Intuitively, points with higher variances (higher uncertainty) contribute less to the optimal depth map. If we solve the resulting optimization problem

$$\min_Z \sum_{i,j \in G} \frac{1}{p_{1ij}} (Z_{ij} - Z_{1ij})^2 + \frac{1}{p_{2ij}} (Z_{ij} - Z_{2ij})^2 \quad (13)$$

we obtain

$$Z_{ij} = \frac{Z_{1ij}/p_{1ij} + Z_{2ij}/p_{2ij}}{1/p_{1ij} + 1/p_{2ij}} \quad (14)$$

This is identical to the update procedure of the Kalman filter (8) if Z_1 is the new measurement Z and Z_2 is the current estimate \hat{Z}^- . In other words, the Kalman filter

update is the solution to a simple surface reconstruction problem. By choosing the weights λ_i to be the inverse variances at every pixel the elementary surface reconstruction problem leads to a Kalman filter update.

This observation suggests that the Kalman filter update stage can be replaced with a surface reconstruction procedure that is modified to take the variances into account and provide for smoothness and discontinuity preservation. In particular, the simple update equation (2) could be replaced by the advanced iterative scheme (5) to recover a smooth depth map.

While this procedure of integrating depth information over time is interesting, the actual acquisition of the data from the image brightness E is not accounted for. This can be done as follows. In vision domains such as motion, stereo and shading functional relationships between measured brightness E and depth Z have been formulated, that can be written in the implicit form

$$f(E, Z) = 0. \quad (15)$$

Examples hereof are given in the following section. Rather than use Z as the measurement input to the temporal surface reconstruction and penalizing for departure from this value, we will penalize directly for the error in the underlying relationship (15) between depth and brightness:

$$\min_{\hat{Z}^+} \sum_{i,j \in G} \frac{1}{p_{ij}} f(E, \hat{Z}_{ij}^+)^2 + \frac{1}{\hat{p}_{ij}} (\hat{Z}_{ij}^+ - \hat{Z}_{ij}^-)^2 + \mu (\nabla \hat{Z}_{ij}^+)^2. \quad (16)$$

where p_{ij} is the variance in the measurement of brightness. The solution to the minimization problem satisfies

$$\frac{f}{p_{ij}} \frac{\partial f}{\partial \hat{Z}^+} + \frac{1}{\hat{p}_{ij}} (\hat{Z}_{ij}^+ - \hat{Z}_{ij}^-) - \mu \Delta Z_{ij} = 0 \quad (17)$$

from which an iterative scheme can be derived as before. However, as we will see, the functional relationship (15) is not usually this simple and each case must be studied individually. A block diagram depicting the generalized temporal surface reconstruction procedure is shown in the bottom diagram of figure 2.

It is straightforward to extend the above procedure to incorporate depth information from different depth sources by adding terms $f_i(E, Z)^2/s$ for each source i to the cost function (16). Another issue is the fact that the variances \hat{p} must be updated in each iteration of the Kalman filter analogous to (9). The description of this process is quite lengthy if smoothness constraints are taken into account. The reader is referred to the description of the underlying Bayesian theory provided in Szeliski [18].

3 Applications of Temporal Surface Reconstruction

In this section we will show how temporal surface reconstruction applies to particular problems in vision. We have selected depth from motion and depth from shading as application domains and provide a brief derivation of the temporal reconstruction equations below.

3.1 Depth from motion

We will consider the case in which a single moving camera acquires a sequence of images from which depth is to be extracted under the assumption of known motion. The images acquired by the camera are denoted by $E(x, y, t)$ where E is the brightness, x, y are the coordinates in the image plane and t is the time.

Horn, Negahdaripour and Weldon [12], [16] use the brightness constancy assumption $\frac{dE}{dt} = 0$ to derive a relationship between image brightness and motion/structure parameters

$$\frac{s \cdot t}{Z} + v \cdot \omega + E_t = 0 \quad (18)$$

This functional is of the form $f(E_x, E_y, E_t, Z) = 0$. The quantities s, v and E_t are computable from the acquired images, the motion t and ω is assumed to be known and Z is the unknown depth. This approach is known as the *direct* method since it links image brightness directly to motion and structure obviating the need for the previously used computationally expensive optical flow.

To avoid the nonlinearity in Z we introduce the quantity $d = 1/Z$ and formulate the update stage of the temporal surface reconstruction in terms of d :

$$\min_d \sum_{i,j \in G} \frac{1}{\rho_{i,j}} ((s_{ij} \cdot t) d_{ij} + v_{ij} \cdot \omega + E_{tij})^2 + \frac{1}{\bar{\rho}_{i,j}} (d_{ij} - \hat{d}_{ij})^2 + \mu (\nabla d_{ij})^2 \quad (19)$$

where the "+" and "-" superscripts have been omitted for simplicity and the hat denotes the estimate obtained from the previous images. The solution to the minimization problem is obtained by the iterative scheme

$$d_{ij}^{(n+1)} = \frac{\mu' \hat{d}_{ij}^{(n)} + \frac{1}{\bar{\rho}_{i,j}} \hat{d}_{ij} - \frac{1}{\rho_{i,j}} (s_{ij} \cdot t) (v_{ij} \cdot \omega + E_{tij})}{\mu' + \frac{1}{\bar{\rho}_{i,j}} + \frac{1}{\rho_{i,j}} (s_{ij} \cdot t)^2} \quad (20)$$

in which \hat{d}_{ij} is a local average of neighbors of d_{ij} which, as the constant μ' depends on the choice of finite differences approximation made for the derivatives.

3.2 Depth from Shading

The image irradiance equation

$$E(x, y) - R(p, q) = 0 \quad (21)$$

is a functional of the form

$$f(E, Z, Z_x, Z_y) = 0 \quad (22)$$

Adding smoothness and integrability constraints (see Horn [13]) to the temporal surface reconstruction we get.

$$\min_{Z, p, q} \sum_{i,j \in G} \frac{[E_{i,j} - R(p_{i,j}, q_{i,j})]^2}{\rho_{i,j}} + \frac{(Z_{i,j} - \hat{Z}_{i,j}^-)^2}{\rho_{i,j}} + \lambda [(\nabla p)^2 + (\nabla q)^2] + \mu [(Z_x - p)^2 + (Z_y - q)^2] \quad (23)$$

Minimization of the above functional leads to three coupled partial differential equations in Z, p and q whose solutions are obtained by the iterative scheme:

$$p^{(n+1)}_{i,j} = \frac{\lambda' \hat{p}_{i,j}^{(n)} + \frac{(E_{i,j} - R(p_{i,j}, q_{i,j}))}{\rho_{i,j}} R_{p,i,j} + \mu Z_{x,i,j}}{\lambda' + \mu} \quad (24)$$

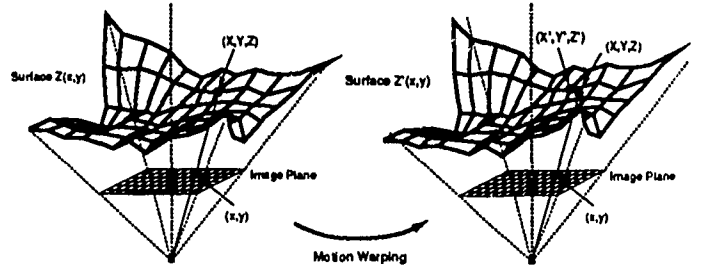


Figure 3: The motion warping.

$$q^{(n+1)}_{i,j} = \frac{\lambda' \hat{q}_{i,j}^{(n)} + \frac{(E_{i,j} - R(p_{i,j}, q_{i,j}))}{\rho_{i,j}} R_{q,i,j} + \mu Z_{y,i,j}}{\lambda' + \mu} \quad (25)$$

$$Z^{(n+1)}_{i,j} = -\frac{\mu' \hat{Z}_{i,j}^{(n)} - \frac{\hat{Z}_{i,j}^-}{\bar{\rho}_{i,j}} - \mu(p_{x,i,j} + q_{y,i,j})}{\mu' + \frac{1}{\bar{\rho}_{i,j}}} \quad (26)$$

Where λ' and μ' are constant multiples of λ and μ respectively. The iterative scheme converges provided the details of the boundary conditions and derivative operators are handled correctly, these issues are discussed in Horn [13].

3.3 - The prediction stage

So far, our description has focused on the update stage of the temporal reconstruction algorithm. As we can see from figure 2 the prediction stage plays the vital role of transforming the depth map estimates Z between iterations to account for the motion of the camera between frames. This is simply a geometric transformation and poses no particular difficulty except for the fact that the surface Z^+ is only known through discrete samples.

We begin with a depth map $Z(x, y)$ with a depth value stored at every point in a discrete grid corresponding to the image array as shown in figure 3.

Each depth map value corresponds to a point $R = [X, Y, Z]^T$ on the surface via the equations of perspective projection. Due to the motion t, ω between frames, such a point will move according to the equations of rigid body motion

$$\dot{R} = -t - \omega \times R \quad (27)$$

so that the resulting surface may appear as in the bottom diagram of figure 3. The difficulty arises when we note that the next measurement which will be used to update the warped surface is available only at the grid point locations and that the discrete points in the warped surface may not project back to those grid point locations. It becomes necessary to resample the warped surface at the grid point locations by interpolating between the given warped samples. This procedure is straightforward, but somewhat lengthy especially when we also consider the variances p^+ in the process. The reader is referred to Heel [9] for details.

4 Experiments

In this section we will illustrate the operation of the temporal surface reconstruction with results obtained on sample images.

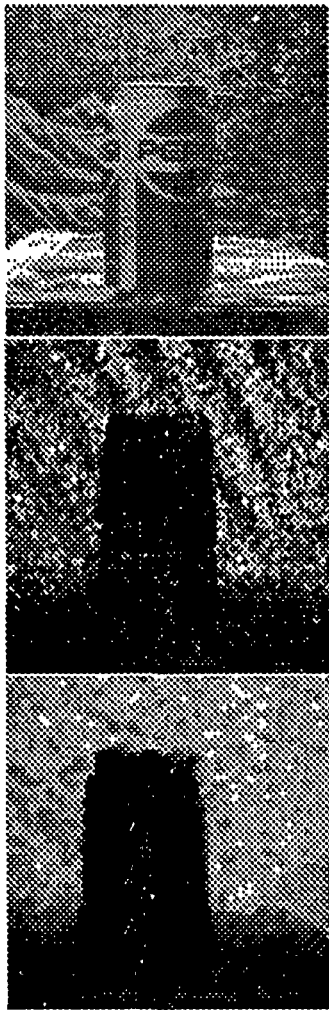


Figure 4: Top: The pepsi scene. Middle: Depth map of a single measurement without reconstruction. Bottom: Depth map after 8 temporal iterations (9 frames).

For the depth from motion algorithm we present results from two real image sequences. An image from the first sequence entitled "pepsi" is shown in figure 4 along with two brightness coded depth maps (brightness is proportional to depth). The middle depth map is the result of simply measuring depth from two images using (18), the bottom depth map shows the result of 8 iterations (9 frames) of the temporal reconstruction algorithm. The camera translated uniformly from left to right during the experiment. In both motion experiments the initial guess for the depth map is a constant (a plane). Since the minimization procedure in each step of the temporal reconstruction procedure can be initialized with the current structure model \hat{Z}_{n-1} only few iterations (10 in this case) are necessary per frame. The noise reduction effect achieved by the algorithm is clearly visible.

Note that the pepsi scene poses a number of difficult problems which have plagued other algorithms in motion vision but are handled nicely by the recursive estimation and surface reconstruction. specularities on the

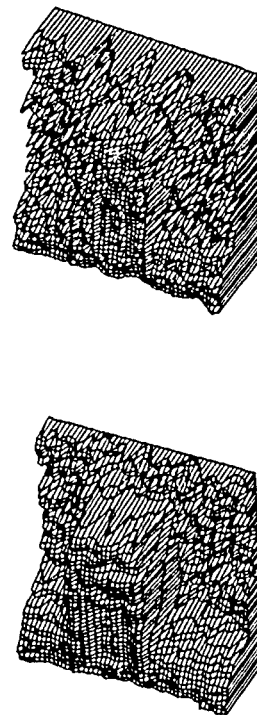


Figure 5: Wire frame plots of the structure from the pepsi scene. Top: after 1 iteration. Bottom: after 8 iterations.

can, large regions of constant brightness underneath the logo. A wire frame structure plot of the recovered depth after 1 and 8 iterations on the pepsi scene is shown in figure 5.

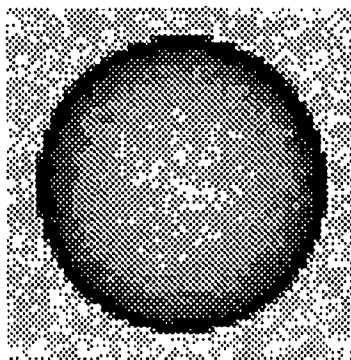
In a second experiment a more complex scene and motion were employed. An image from the "cup" scene in which the camera translates $t = [2, 0, 4]$ mm between frames is shown in figure 6. The recovered depth map after 8 iterations is shown in the bottom diagram of figure 6. Note that the algorithm detects the spoon that was placed in the cup but is barely visible in the image.

To test the shape from shading algorithm lambertian shaded shapes were used. Successive overrelaxation with $\alpha = -1.7$ and linearization of the reflectance map were in effect. The motion between frames was uniform translation in both x and y . Brightness values are in the range $[0, 1]$ and Gaussian noise of variance 0.05 was added to each pixel. Random numbers were used as initial guesses for the depth map. Figure 7 shows the top view of a hemisphere which translates diagonally during the sequence along the development of the root mean squared error of the depth values (with respect to the true depth values) as a function of the frame number.

Figure 8 compares the actual input depth on the top with the recovered structure after 8 iterations of the algorithm. Since the estimate from the previous iteration can be used to initialize the minimization in the next time step, only 50 iterations are necessary for each frame as opposed to several thousand in the static depth from shading algorithm. As multiple frames are used to re-



Figure 6: Top: An image from the cup sequence. Bottom: recovered depth map after 8 iterations.



RMSE Error in Depth vs. Frame Number
Sphere Sequence

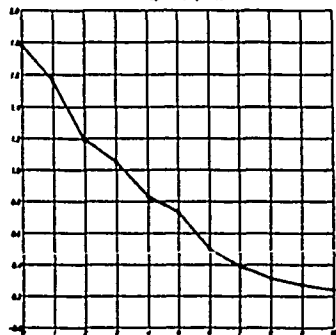


Figure 7: Top: Image from the sphere sequence with 5 noise. Bottom: Root mean squared error over frame number for temporal reconstruction.

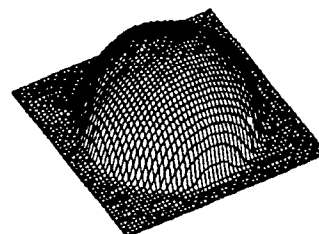
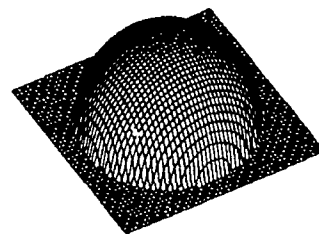


Figure 8: Top: wire frame plot of true depth in the sphere sequence. Bottom: wire frame plot of depth recovered after 8 iterations.

duce the effect of noise, higher accuracy can be obtained than with a repeated application of the static algorithm.

5 Conclusion

This paper has presented a method for the temporal integration of surface reconstruction. This algorithm combines the features of recursive estimation theory and surface reconstruction and is capable of aggregating depth information from different sources into one continuously improving structure model of the scene. The effectiveness of this method has been demonstrated for different domains on a variety of images. Combination of different sources has not been demonstrated here but is achieved in a straightforward manner and will be the focus of our future research.

Acknowledgements

We would like to thank Davi Geiger, Richard Szeliski and Berthold Horn for valuable discussions of the ideas in this paper.

References

- [1] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, 1987.
- [2] R. C. Bolles and H. H. Baker. Epipolar-plane image analysis: A technique for analyzing motion sequences. In *IEEE Proceedings of the Third Work-*

- shop on Computer Vision: Representation and Control, Bellaire, MI, October 1985.
- [3] T. J. Broida and R. Chellappa. Kinematics and structure of a rigid object from a sequence of noisy images. In *Proceedings of the IEEE Workshop on Motion: Representation and Analysis*, Charleston, SC, May 1986.
 - [4] D. Geiger and F. Girosi. Parallel and deterministic algorithms for mrfs: surface reconstruction and integration. AI Memo 1114, MIT Artificial Intelligence Laboratory, June 1989.
 - [5] A. Gelb (Ed.). *Applied Optimal Estimation*. The MIT Press, Cambridge, MA, 1974.
 - [6] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6, 1984.
 - [7] W. E. L. Grimson. *From Images to Surfaces*. The MIT Press, 1981.
 - [8] J. Heel. Dynamic motion vision. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, CA, May 1989.
 - [9] J. Heel. Direct estimation of structure and motion from multiple frames. AI Memo 1190, MIT Artificial Intelligence Laboratory, March 1990.
 - [10] B. K. P. Horn. *Robot Vision*. The MIT Press, 1986.
 - [11] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17, 1981.
 - [12] B. K. P. Horn and E. J. Weldon, Jr. Direct methods for recovering motion. *International Journal of Computer Vision*, 2, 1988.
 - [13] B. K. P. Horn. Height and Gradient from Shading. AI Memo 1105, MIT Artificial Intelligence Laboratory, May 1989.
 - [14] J. L. Marroquin. Deterministic bayesian estimation of markovian random fields with applications to computational vision. In *Proceedings of the International Conference on Computer Vision*, London, England, June 1987.
 - [15] L. Matthies, R. Szeliski, and R. Kanade. Incremental estimation of dense depth maps from image sequences. In *Proceedings Computer Vision and Pattern Recognition*, Ann Arbor, MI, June 1988.
 - [16] S. Negahdaripour and B. K. P. Horn. A direct method for locating the focus of expansion. *Computer Vision, Graphics and Image Processing*, Vol. 46(3), June 1989.
 - [17] T. Poggio, E. Gamble, and J. Little. Parallel integration of vision modules. *Science*, Vol. 242, October 1988.
 - [18] R. S. Szeliski. Bayesian modeling of uncertainty in low-level vision. Technical Report CMU-CS-88-169, Computer Science Department, Carnegie Mellon University, August 1988.
 - [19] D. Terzopoulos. Multilevel computational processes for visual surface reconstruction. *Computer Vision, Graphics and Image Processing*, 24, 1983.
 - [20] S. Ullman. Maximizing rigidity: The incremental recovery of 3-d structure from rigid and rubbery motion. AI Memo 721, MIT Artificial Intelligence Laboratory, June 1983.
 - [21] M. Yamamoto. The image sequence analysis of three-dimensional dynamic scenes. Technical Report UDC 681.3.056, Electrotechnical Laboratory, Agency of Industrial Science and Technology, Japan, May 1988.

A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment

Takeo Kanade and Masatoshi Okutomi
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

A central problem in stereo matching by computing correlation or sum of squared differences (SSD) lies in selecting an appropriate window size. If the window is too small and does not cover enough intensity variation, it gives a poor disparity estimate, because the signal (intensity variation) to noise ratio is low. If, on the other hand, the window is too large and covers a region in which the depth of scene points varies, then the disparity within the window is not constant. As a result, the position of maximum correlation or minimum SSD may not represent a correct estimate of disparity. For this reason, an appropriate window size must be selected locally. There has been, however, little research directed toward the adaptive selection of matching windows.

The stereo algorithm we propose in this paper selects a window adaptively by evaluating the local variation of the intensity *and* the disparity. We employ a statistical model that represents uncertainty of disparity of points over the window: the uncertainty is assumed to increase with the distance of the point from the center point. This modeling enables us to assess how disparity variation within a window affects the estimation of disparity. As a result, we can compute the uncertainty of the disparity estimate which takes into account both intensity and disparity variances. So, the algorithm can search for a window that produces the estimate of disparity with the least uncertainty for each pixel of an image. The method controls not only the size but also the shape (rectangle) of the window. The algorithm has been tested on both synthetic and real images, and the quality of the disparity maps obtained demonstrates the effectiveness of the algorithm.

1 Introduction

Stereo matching by computing correlation or sum of squared differences (SSD) is a basic technique for obtaining a dense depth map from images [Matthies *et al.*, 1989][Forstner and Pörtl, 1986][Wood, 1983][Mori *et al.*, 1973]. A central problem with this method lies in selecting an appropriate window size. If the window is too small and does not cover enough intensity variation, it gives a poor disparity estimate, because

the signal (intensity variation) to noise ratio is low. If, on the other hand, the window is too large and covers a region in which the depth of scene points varies, then the disparity within the window is not constant. Therefore, the position of maximum correlation or minimum SSD may not represent a correct estimate of disparity. For this reason, an appropriate window size must be selected locally.

However, there has been little research for adaptive window selection. Most correlation- or SSD-based methods in the past have used a window of a fixed size that is chosen empirically for each application. Uncertainty in matching due to the variation of unknown disparities within a window is unaccounted for by existing stereo algorithms. Levine *et al.* [Levine *et al.*, 1973] presented a method of changing the window size locally depending on only the intensity pattern. However, window selection must also depend on the disparity (i.e. depth) variations which changes from pixel to pixel in an image. In fact, the difficulty in obtaining an adaptive window lies in a difficulty in evaluating and using disparity variances. While the intensity variation is directly obtained from the image, evaluation of the disparity variation is not easy, since the disparity is what we intend to calculate as an end product of stereo. To resolve the dilemma, an appropriate model of disparity variation is required which enables us to assess how disparity variation within a window affects the estimation of disparity.

The stereo algorithm we propose in this paper selects a window adaptively by evaluating the local variation of the intensity and the disparity. We employ a statistical model that represents uncertainty of disparity of points over the window: the uncertainty is assumed to increase with the distance of the point from the center point. This modeling enables us to compute both a disparity estimate *and* the uncertainty of the estimate. So, the algorithm can search for a window that produces the estimate of disparity with the least uncertainty for each pixel of an image. The method controls not only the size but also the shape (rectangle) of the window.

In this paper, we first develop a model of stereo matching in section 2. Section 3 shows how to estimate the most likely disparity and the uncertainty of the estimate based on the modeling in section 2. These two sections provide theoretical grounds of our proposed algorithm. In section 4, we present a complete stereo algorithm which selects appropriate window size and shape adaptively for each pixel. Section 5 provides experimental results with real stereo images. The quality of the disparity maps obtained demonstrates the effectiveness of

the algorithm.

2 Modeling Stereo Matching

We will first develop a statistical model of the difference of intensities of two images within a window. The analysis is based on the uncertainty model presented in [Okutomi and Kanade, 1990b]. Let the stereo intensity images be $f_1(x, y)$ and $f_2(x, y)$. Assume that the baseline is parallel to the x axis, and $f_1(x, y)$ and $f_2(x, y)$ come from an underlying intensity function $f(x, y)$ with a disparity function $d_r(x, y)$. Then,

$$f_1(x, y) = f(x, y) + n_1(x, y) \quad (1)$$

$$f_2(x + d_r(x, y), y) = f(x, y) + n_2(x, y), \quad (2)$$

where $n_1(x, y)$ and $n_2(x, y)$ are independent Gaussian white noise for both images, such that

$$n_1(x, y), n_2(x, y) \sim N(0, \sigma_n^2). \quad (3)$$

From equations (1) and (2),

$$f_1(x, y) - f_2(x + d_r(x, y), y) = n(x, y), \quad (4)$$

where $n(x, y)$ is Gaussian white noise such that

$$n(x, y) \sim N(0, 2\sigma_n^2). \quad (5)$$

To simplify the notation, suppose that we want to compute the disparity at $(x, y) = (0, 0)$, i.e., the value $d_r(0, 0)$. Also, suppose a window $W = \{(\xi, \eta)\}$ is placed at the correct corresponding positions in both images, that is, at $(0, 0)$ in image $f_1(x, y)$ and at $(d_r(0, 0), 0)$ in image $f_2(x, y)$. Figure 1 illustrates the situation. Then, the difference of intensities between f_1 and f_2 at (ξ, η) in the window can be approximated by using the Taylor expansion of the left hand side of equation (4)

$$f_1(\xi, \eta) - f_2(\xi + d_r(0, 0), \eta) \approx (d_r(\xi, \eta) - d_r(0, 0)) \frac{\partial}{\partial \xi} f_2(\xi + d_r(0, 0), \eta) + n(\xi, \eta). \quad (6)$$

At this point, let us introduce the following statistical model for the disparity $d_r(\xi, \eta)$ within a window:

$$d_r(\xi, \eta) - d_r(0, 0) \sim N\left(0, \alpha_d \sqrt{\xi^2 + \eta^2}\right), \quad (7)$$

where α_d is a constant that represents the amount of fluctuation of the disparity. That is, this model assumes that the difference of disparity at a point (ξ, η) in the window from that of the center point $(0, 0)$ has a zero-mean Gaussian distribution with variance proportional to the distance between these points. In other words, the expected value of the disparity at (ξ, η) is the same as the center point, but it is expected to fluctuate more as the point is farther from the center.¹ Or, in terms of the scene, the surface covered by the window is expected to be locally flat

and parallel to the baseline, but it is less certain as the window becomes larger. We also assume that the image intensity derivatives $\frac{\partial}{\partial \xi} f_2(\xi, \eta)$ within a window follow a zero-mean Gaussian white distribution,² and that intensity derivatives $\frac{\partial}{\partial \xi} f_2(\xi, \eta)$ and disparities $d_r(\xi, \eta)$ are mutually independent.

These assumptions allow us to model a statistical distribution of the intensity difference (7). Let us denote the right hand side of equation (7) by $n_s(\xi, \eta)$. First, we compute the mean and variance of $n_s(\xi, \eta)$:

$$\begin{aligned} E[n_s(\xi, \eta)] &= E[d_r(\xi, \eta) - d_r(0, 0)] E\left[\frac{\partial}{\partial \xi} f_2(\xi + d_r(0, 0), \eta)\right] \\ &\quad + E[n(\xi, \eta)] = 0 \end{aligned} \quad (8)$$

$$\begin{aligned} E[(n_s(\xi, \eta))^2] &= E\left[\left((d_r(\xi, \eta) - d_r(0, 0)) \frac{\partial}{\partial \xi} f_2(\xi + d_r(0, 0), \eta)\right)^2\right] \\ &\quad + E\left[2(d_r(\xi, \eta) - d_r(0, 0)) \cdot \left(\frac{\partial}{\partial \xi} f_2(\xi + d_r(0, 0), \eta)\right) n(\xi, \eta)\right] \\ &\quad + E[(n(\xi, \eta))^2] \\ &= E[(d_r(\xi, \eta) - d_r(0, 0))^2] \cdot E\left[\left(\frac{\partial}{\partial \xi} f_2(\xi + d_r(0, 0), \eta)\right)^2\right] + E[(n(\xi, \eta))^2] \\ &= 2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi^2 + \eta^2}, \end{aligned} \quad (9)$$

where

$$\alpha_f = E\left[\left(\frac{\partial}{\partial \xi} f_2(\xi + d_r(0, 0), \eta)\right)^2\right]. \quad (10)$$

Appendix I shows that $n_s(\xi, \eta)$ is white noise and its distribution can be approximated by a Gaussian distribution with the above mean and variance. That is,

$$\begin{aligned} n_s(\xi, \eta) &\approx f_1(\xi, \eta) - f_2(\xi + d_r(0, 0), \eta) \\ &\sim N\left(0, 2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi^2 + \eta^2}\right). \end{aligned} \quad (11)$$

The intuitive interpretation of (11) is as follows. Referring to figure 1, $n_s(\xi, \eta)$ is the difference between f_1 and f_2 at (ξ, η) within a window when the window is placed at the corresponding positions for obtaining the disparity at $(0, 0)$. If there is no additive noise $n(x, y)$ in the image (i.e., $\sigma_n = 0$) and the disparity is constant within the window (i.e., $\alpha_d = 0$), then the two images match exactly, and $n_s(\xi, \eta)$ must be null. Otherwise, however, the difference has a value which shows a combined noise characteristic which comes from both intensity and disparity variations. As derived in (11), it can be modeled by zero-mean Gaussian noise whose variance is

²This is also equivalent to assuming the pattern $f_2(\xi, \eta)$ to be result of Brownian motion: i.e., locally it has a constant brightness, but has more fluctuation as the window becomes bigger.

¹The statistical model of (7) can be shown equivalent to assuming that $d_r(\xi, \eta)$ is generated by Brownian motion (refer to [B.B.Mandelbrot and Ness, 1968][Voss, 1987]). More generally, we can assume $d_r(\xi, \eta)$ to be a fractal. This corresponds to choosing a different degree of $\xi^2 + \eta^2$ in the variance in (7). The Brownian motion is the simplest case in which the degree is $\frac{1}{2}$. However, our preliminary experiments have shown no noticeable advantage of using a general fractal assumption.

a summation of a constant term and a term proportional to $\sqrt{\xi^2 + \eta^2}$. The constant term is from the noise added to the image intensities. The second term is from *uncertain local support*. That is, while the points surrounding the center point in the window are used to support the matching for the center point, it should be noted that these points may actually increase the error in computing the disparity of the center point. This is because, in general, the disparity of the surrounding points deviates from that of the center point. This uncertainty is represented as if the intensity signals have additional noise whose power is proportional to the distance from the center point in the window. If the disparity is constant over the window (i.e. $\alpha_d = 0$), the additional noise is zero. If the disparity changes more in the window (i.e., the larger α_d is), its effect becomes larger and the information contributed by the surrounding points becomes more uncertain.

3 Estimating Disparity and Its Uncertainty

Now, we will show how the disparity and its uncertainty can be estimated based on the modeling presented in the previous section. Let $d_0(x, y)$ be an initial estimate of the disparity $d_r(x, y)$. By using the Taylor expansion, equation (11) becomes

$$n_s(\xi, \eta) = f_1(\xi, \eta) - f_2(\xi + d_0(0, 0), \eta) - \Delta d \frac{\partial}{\partial \xi} f_2(\xi + d_0(0, 0), \eta), \quad (12)$$

where Δd is an incremental correction of the estimate to be made, such that $\Delta d = d_r(0, 0) - d_0(0, 0)$. Dividing both sides of this equation by $\sqrt{2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi^2 + \eta^2}}$ yields

$$n_n(\xi, \eta) = \frac{f_1(\xi, \eta) - f_2(\xi + d_0(0, 0), \eta) - \Delta d \frac{\partial}{\partial \xi} f_2(\xi + d_0(0, 0), \eta)}{\sqrt{2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi^2 + \eta^2}}}, \quad (13)$$

where $n_n(\xi, \eta) = \frac{n_s(\xi, \eta)}{\sqrt{2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi^2 + \eta^2}}}$ is Gaussian white noise such that

$$n_n(\xi, \eta) \sim N(0, 1). \quad (14)$$

By letting

$$\phi_1(\xi, \eta) = \frac{f_1(\xi, \eta) - f_2(\xi + d_0(0, 0), \eta)}{\sqrt{2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi^2 + \eta^2}}} \quad (15)$$

$$\phi_2(\xi, \eta) = \frac{\frac{\partial}{\partial \xi} f_2(\xi + d_0(0, 0), \eta)}{\sqrt{2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi^2 + \eta^2}}}, \quad (16)$$

we have

$$\phi_1(\xi, \eta) - \Delta d \phi_2(\xi, \eta) = n_n(\xi, \eta). \quad (17)$$

Now, by sampling ϕ_1 and ϕ_2 at (ξ_i, η_j) in the window W we can define ξ_{ij} as

$$\xi_{ij} = \phi_1(\xi_i, \eta_j) - \Delta d \phi_2(\xi_i, \eta_j) = n_n(\xi_i, \eta_j). \quad (18)$$

From equation (14), the conditional probability density function of ξ_{ij} given Δd is

$$p(\xi_{ij}|\Delta d) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(\phi_1(\xi_i, \eta_j) - \Delta d \phi_2(\xi_i, \eta_j))^2}{2}\right). \quad (19)$$

Since $n_n(\xi, \eta)$ is white noise, ξ_{ij} are mutually independent. So we get

$$p(\xi_{ij}(i, j \in W)|\Delta d) = \prod_{i,j \in W} p(\xi_{ij}|\Delta d), \quad (20)$$

where $p(\xi_{ij}(i, j \in W)|\Delta d)$ is the conditional joint probability for the points in the window, and $\prod_{i,j \in W}$ denotes the product over the window. From the continuous version of Bayes' theorem,

$$p(\Delta d|\xi_{ij}(i, j \in W)) = \frac{p(\xi_{ij}(i, j \in W)|\Delta d)p(\Delta d)}{\int_{-\infty}^{\infty} p(\xi_{ij}(i, j \in W)|\Delta d)p(\Delta d)d(\Delta d)}. \quad (21)$$

Assuming no prior information of Δd (i.e., $p(\Delta d) = \text{constant}$), substitution of (19) into (21) yields

$$p(\Delta d|\xi_{ij}(i, j \in W)) = \frac{1}{\sqrt{2\pi}\sigma_{\Delta d}} \exp\left(-\frac{(\Delta d - \hat{\Delta d})^2}{2\sigma_{\Delta d}^2}\right), \quad (22)$$

where

$$\hat{\Delta d} = \frac{\sum_{i,j \in W} (\phi_1(\xi_i, \eta_j) \phi_2(\xi_i, \eta_j))}{\sum_{i,j \in W} (\phi_2(\xi_i, \eta_j))^2} \quad (23)$$

$$\sigma_{\Delta d}^2 = \frac{1}{\sum_{i,j \in W} (\phi_2(\xi_i, \eta_j))^2}, \quad (24)$$

where $\sum_{i,j \in W}$ denotes the summation over the window. Or, by substituting equations (15) and (16) into equations (23) and (24), we obtain

$$\hat{\Delta d} = \frac{\sum_{i,j \in W} \frac{(f_1(\xi_i, \eta_j) - f_2(\xi_i + d_0(0, 0), \eta_j)) \frac{\partial}{\partial \xi} f_2(\xi_i + d_0(0, 0), \eta_j)}{2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi_i^2 + \eta_j^2}}}{\sum_{i,j \in W} \frac{(\frac{\partial}{\partial \xi} f_2(\xi_i + d_0(0, 0), \eta_j))^2}{2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi_i^2 + \eta_j^2}}} \quad (25)$$

$$\sigma_{\Delta d}^2 = \frac{1}{\sum_{i,j \in W} \frac{(\frac{\partial}{\partial \xi} f_2(\xi_i + d_0(0, 0), \eta_j))^2}{2\sigma_n^2 + \alpha_f \alpha_d \sqrt{\xi_i^2 + \eta_j^2}}}. \quad (26)$$

Equation (22) says that the conditional probability density function of Δd given the observed stereo intensities over the window becomes a Gaussian probability density function. The mean value and the variance of the Gaussian probability are $\hat{\Delta d}$ and $\sigma_{\Delta d}^2$, computed with equations (25) and (26). That is, $\hat{\Delta d}$ and $\sigma_{\Delta d}^2$ provide the maximum likelihood estimate of the disparity (increment) and the uncertainty of the estimation for the given window W , respectively.

α_d and α_f are parameters that represent the disparity fluctuation and the intensity fluctuation, respectively. We estimate them locally within the window from equations (7) and (10),

$$\alpha_d = \frac{1}{N_w} \sum_{i,j \in W} \frac{(d_0(\xi_i, \eta_j) - d_0(0, 0))^2}{\sqrt{\xi_i^2 + \eta_j^2}} \quad (27)$$

$$\alpha_f = \frac{1}{N_w} \sum_{i,j \in W} \left(\frac{\partial}{\partial \xi} f_2(\xi_i + d_0(0, 0), \eta_j) \right)^2, \quad (28)$$

where N_w is the number of the samples within the window. These parameters change as the shape and size of a window changes.

4 Iterative Stereo Algorithm with an Adaptive Window

In the previous sections we have developed a theory for computing the estimates of the disparity increment and its uncertainty, which take into account the fact that not only the intensity but also the disparity varies within a window. We now describe the complete stereo algorithm based on the theory:

1. Start with an initial disparity estimate $d_0(x, y)$. This initial estimate can be obtained by any existing stereo algorithm.
2. For each point (x, y) , choose a window that provides the estimate of disparity increment having the lowest uncertainty. For the chosen window, calculate the disparity increment by (25) and update the disparity estimate by $d_{i+1}(x, y) = d_i(x, y) + \Delta d(x, y)$.

Here we need a strategy to select a window that results in the disparity estimate having the lowest uncertainty. In the discussions so far the shape of the window can be arbitrary. In practice we limit ourselves to a rectangular window, as illustrated in figure 2, whose width and height can be independently controlled in all four directions. Our strategy is as follows:

- (a) Place a small 3×3 window centered at the pixel, and compute the uncertainty by using (27), (28), and (26).
- (b) Expand the window by one pixel in one direction, e.g., to the right $x+$, for trial, and compute the uncertainty for the expanded window. If the expansion increases the uncertainty, the direction is prohibited from further expansions. Repeat the same process for each of the four directions $x+$, $x-$, $y+$, and $y-$ (excluding the already prohibited ones).
- (c) Compare the uncertainties for all the directions tried and choose the direction which produces the minimum uncertainty.
- (d) Expand the window by one pixel in the chosen direction.
- (e) Iterate steps (b) to (d) until all directions become prohibited from expansion or until the window size reaches to a limit that is previously set.

Thus, our strategy is basically a sequential search for the best window by maximum descent starting with the smallest window

3. Iterate the above process until the disparity estimate $d_i(x, y)$ converges, or up to a certain maximum number of iterations.

Now, by using synthesized data we will examine how the window is adaptively set by the stereo algorithm for each position in an image, and demonstrate its advantage. Figures 3 (a) and (b) show the left and the right images of the test data. In generating the data set, a linear ramp in the direction of the baseline is used as the underlying intensity pattern. It is deformed according to the disparity pattern in figures 3 (c) and (d), and Gaussian noise is added independently to both images. We apply the iterative stereo algorithm to the resultant data.

First, we will examine the result of window selection. The four images in figure 4 show the length (increasing brightness corresponds to increasing length) by which the window has been extended in each of the four directions.³ For example, the vertical dark stripes in figure 4 (a) on the right hand side of the vertical disparity edge show that the windows for those points are not extended to the left so that the windows do not cross the disparity edge to a region of different disparity. We observe the same phenomena in the other directions. We can examine the size and shape of selected windows at several representative positions shown in figure 5. The windows selected at those positions are drawn by dashed lines in figure 6 relative to the disparity edges drawn by solid lines. For example, at $P0$ a window has been expanded to the limit for all directions, whereas at $P1$ expansion to the right has been stopped at the disparity edge. At $P5$, a window is elongated either vertically or horizontally, depending on the image noise, but consistently avoids the corner of the disparity jump.

Next, let us examine the computed disparities. For comparison, we also have computed disparities by running the same iterative algorithm but with a fixed window size; that is, in Step 2 of the stereo algorithm we use a window of predetermined size rather than the window selection strategy. We run with three window sizes, 3×3 , 7×7 , and 15×15 . Figures 7 (a), (b) and (c) show the result produced by fixed window sizes, and (d) by the adaptive-window algorithm. We can clearly see the problem with using a predetermined fixed window size. A larger window is good for flat surfaces, but it blurs the disparity edges. In contrast, a smaller window gives sharper disparity edges at the expense of noisy surfaces. The computed disparity by the proposed algorithm shown in figure 7 (d) shows both smooth flat surfaces and sharp disparity edges. The improvements are further visible by plotting the absolute difference between the computed and true disparities as shown in figure 8, with a table that lists their mean error values. The adaptive-window algorithm has the smallest mean error, but more importantly we should observe that the algorithm has reduced two types of errors. A small fixed window results in large random error everywhere. A large fixed window removes the random error, but introduces systematic errors along the disparity edges. The adaptive-window based method generates small errors of both types. In fact, we have shown that at *each* point the expected value of the error by the adaptive-window method is always smaller than or equal to that produced when *any* fixed-size window is used [Okutomi and Kanade, 1990b].

Figures 9 (a) and (b) show another example of synthesized test data. Figure 10 presents the computed disparity by the new method in (d), together with the results produced by fixed window sizes in (a) to (c) for comparison. As with the previous example, we clearly see better performance with the new method.

5 Experimental Results

We have applied the adaptive-window based stereo matching algorithm presented in this paper to real stereo images.

Figure 11 shows images of a town model that were taken by moving the camera vertically. The disparity, therefore, is

³Actually these are the average of ten runs with different noises to obtain the general tendency, rather than accidental set up.

in the vertical direction. To give an idea of the arrangement of objects in the scene, a picture taken from an oblique angle is given in figure 11 (c).

For initial disparity estimates, we have used a technique of multiple-baseline stereo matching [Okutomi and Kanade, 1990a] which can remove matching ambiguities due to repetitive patterns, especially in the top portion of the image. Figure 12 (a) shows the disparity map computed by the adaptive window algorithm. In addition, the uncertainty estimate computed by the algorithm is shown in figure 12 (b): increasing brightness corresponds to higher uncertainty. With this uncertainty estimate we can locate the regions whose computed disparity is not very reliable (very white regions in figure 12 (b)). In this example, they are either due to aliasing caused by the fine texture of roof tiles of a building (in the middle part of the image) or due to occlusion (the others). The disparity estimates of those uncertain parts can be discarded for later processing. The isometric plot of the disparity map is shown in figure 12 (d), which roughly corresponds to the viewing angle of figure 11 (c). We can see that each building wall has a smooth surface and yet is clearly separated from others, and the shape of the distant bridge (on the left) is recovered.

Figure 13 shows perspective views of the recovered scene by texture mapping the original intensity image on the constructed depth map and generating views from new positions which are outside of the original stereo views. They can give an idea of the quality of reconstruction. This stereo data set is the same one used in [Matthies *et al.*, 1989]. We can observe a noticeable improvement of the result over the previous result. Also it should be noted that this is extremely narrow baseline stereo: the baseline is only 1.2 cm long and the scene is about 1m away from the camera, thus the depth to the baseline ratio is approximately 80.

Figures 14 (a) and (b) show another set of real stereo images which are top views of a coal mine model. Figures 15 (a) and (c) show the isometric plots of the computed disparity. For comparison, actual pictures of the model taken from roughly the same angles are given in figures 15 (b) and (d). The shapes of buildings, a \wedge -shaped roof, a water tank on the roof, and a flat ground have been recovered without blurring edges.

6 Conclusions

In this paper, we have presented an iterative stereo matching algorithm using an adaptive window. The algorithm selects a window adaptively for each pixel. The selected window is optimal in the sense that it produces the disparity estimate having the least uncertainty. By evaluating both the intensity and the disparity variations within a window, we can compute both the disparity estimate and its uncertainty which can then be used for selecting the optimal window.

The key idea for the algorithm is that it employs a statistical model that represents uncertainty of disparity of points over the window: the uncertainty is assumed to increase with the distance of the point from the center point. This model has enabled us to assess how disparity variation within a window affects the estimation of disparity.

An important feature of the algorithm is that it is completely local and does not include any global optimization. Also, the algorithm does not use any post-processing smoothing, but smooth surfaces are recovered as smooth while sharp disparity edges are retained.

The experimental results have demonstrated a clear advantage of this algorithm over algorithms with a fixed-size window both on synthetic and on real images.

References

- [Barnard, 1980] S. T. Barnard. Disparity analysis of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(4):333-340, July 1980.
- [B.B.Mandelbrot and Ness, 1968] B.B.Mandelbrot and B. Ness. Fractional brownian motion, fractional noises and applications. *SIAM*, 10(4):422-438, 1968.
- [de Coulon, 1986] F. d. Coulon. *Signal Theory and Processing*. Artech House, Inc., 1986.
- [Forstner and Pertl, 1986] W. Forstner and A. Pertl. *Photogrammetric Standard Methods and Digital Image Matching Techniques for High Precision Surface Measurements*, pages 57-72. Elsevier Science Publishers B.V., 1986.
- [Grimson, 1985] W. E. L. Grimson. Computational experiments with a feature based stereo algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(1):17-34, January 1985.
- [Horn and B.G.Schunck, 1981] B. Horn and B.G.Schunck. Determining optical flow. *Artificial Intelligence*, 17:185-203, 1981.
- [Levine *et al.*, 1973] M. D. Levine, D. A. O'Handley, and G. M. Yagi. Computer determination of depth maps. *Computer Graphics and Image Processing*, 2(4):131-150, 1973.
- [Mallat, 1988] S. Mallat. Multiresolution representations and wavelets. *PhD thesis, University of Pennsylvania*, 1988.
- [Marr and Poggio, 1977] D. Marr and T. Poggio. A theory of human stereo vision. In *AI Memo*, volume 451. MIT, 1977.
- [Matthies and Okutomi, 1989] L. Matthies and M. Okutomi. A bayesian foundation for active stereo vision. In *SPIE, Sensor Fusion II: Human and Machine Strategies*, November 1989.
- [Matthies *et al.*, 1989] L. Matthies, R. Szeliski, and T. Kanade. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3:209-236, 1989.
- [Mayhew and Frisby, 1981] J. E. Mayhew and J. P. Frisby. Psychophysical and computational studies towards a theory of human stereopsis. *Artificial Intelligence*, 17:349-385, 1981.
- [Mori *et al.*, 1973] K. Mori, M. Kidode, and H. Asada. An iterative prediction and correction method for automatic stereo-comparison. *Computer Graphics and Image Processing*, 2:393-401, 1973.
- [Ohta and Kanade, 1985] Y. Ohta and T. Kanade. Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(2):139-154, March 1985.
- [Okutomi and Kanade, 1990a] M. Okutomi and T. Kanade. A multiple-baseline stereo matching algorithm. *in preparation*, 1990.

[Okutomi and Kanade, 1990b] M. Okutomi and T. Kanade. A signal matching algorithm: An adaptive window based on a brownian motion. Technical Report in preparation, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, 1990.

[Voss, 1987] R. F. Voss. Fractals in nature. In *Course note on FRACTALS: Introduction, Basics, and Perspectives*, 1987.

[Witkin *et al.*, 1987] A. Witkin, D. Terzopoulos, and M. Kass. Signal matching through scale space. *International Journal of Computer Vision*, pages 133–144, 1987.

[Wood, 1983] G. Wood. Realities of automatic correlation problem. *Photogrammetric Engineering and Remote Sensing*, 49:537–538, April 1983.

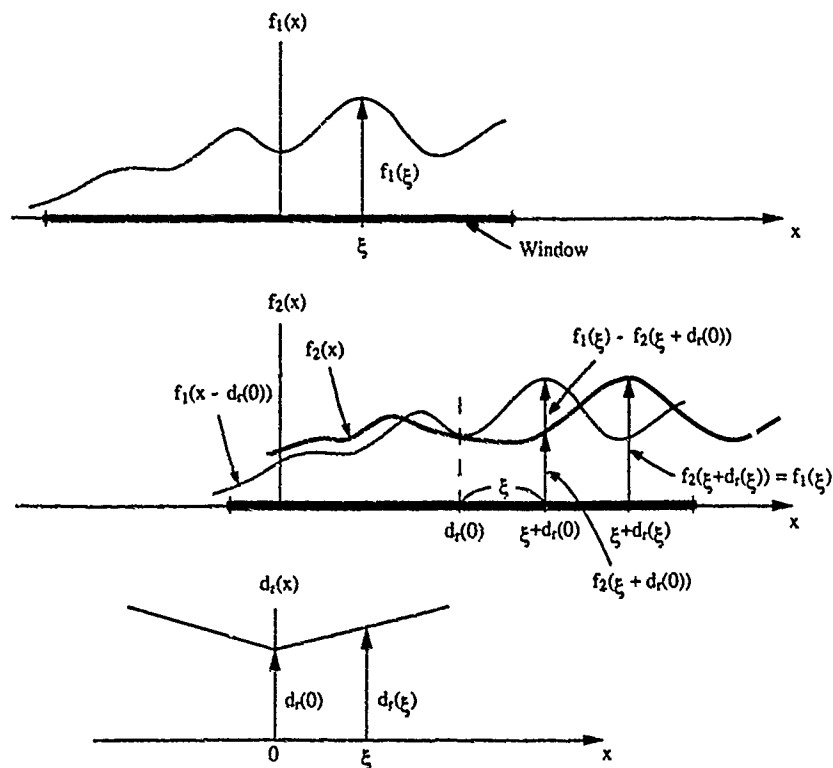


Figure 1: Illustration of $n_s(\xi, \eta)$ in one dimension. The graph at the top shows $f_1(x)$; the middle one, $f_2(x)$ (the thicker curve) with $f_1(x)$ shifted by $d_r(0)$ (the thinner curve); the bottom one, $d_r(x)$. The region indicated by the very thick lines on the axes indicate the region covered by the window.

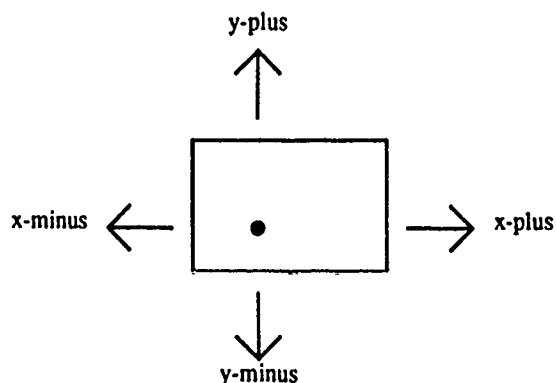


Figure 2: Window expansion

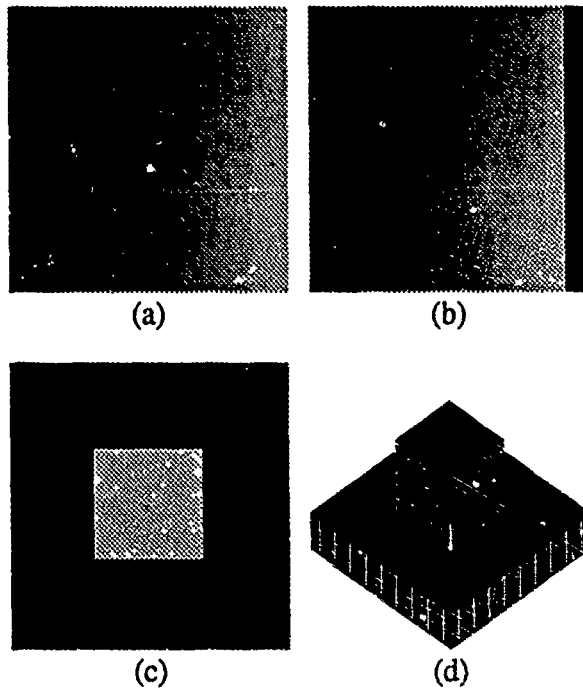


Figure 3: Synthesized stereo images, with a ramp intensity pattern with Gaussian noise: (a) Left image; (b) Right image; (c) Disparity pattern; (d) An isometric plot of the disparity pattern

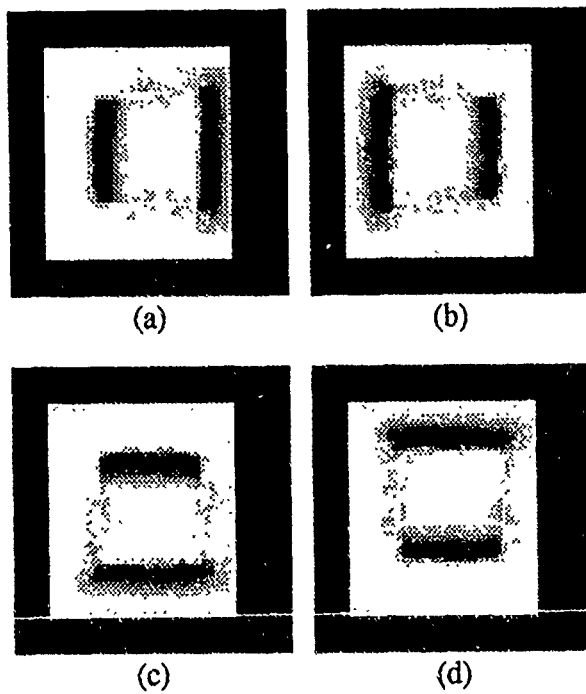


Figure 4: Extent of window-size expansion for each direction: (a) Left (X-minus) direction; (b) Right (X-plus) direction; (c) Down (Y-minus) direction (d) Up (Y-plus) direction

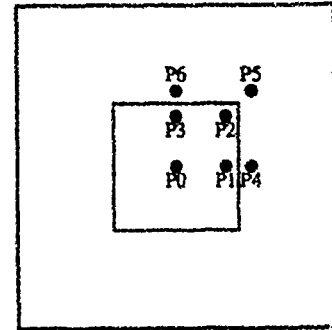


Figure 5: Positions for which size and shape of selected windows are examined.

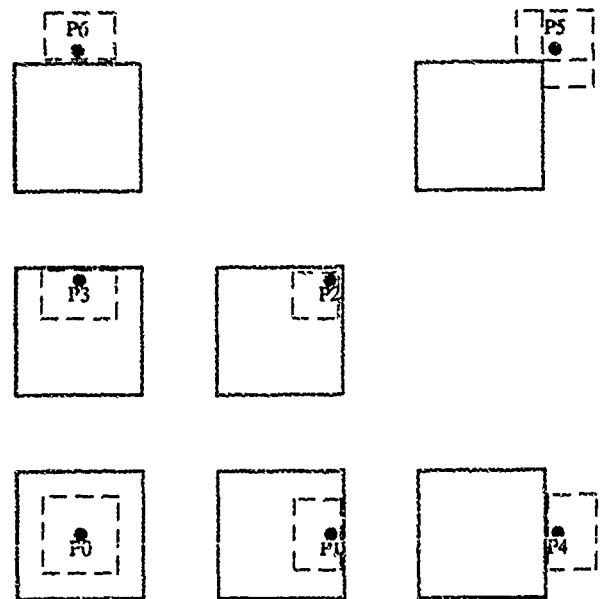


Figure 6: Selected windows for each position

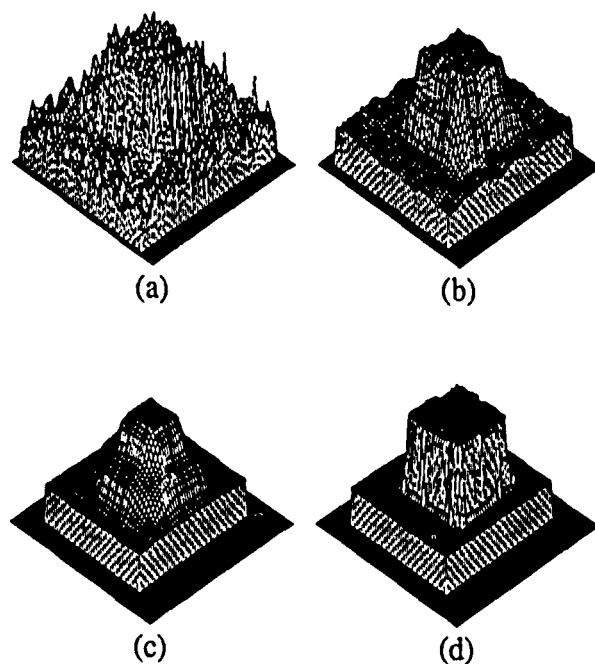
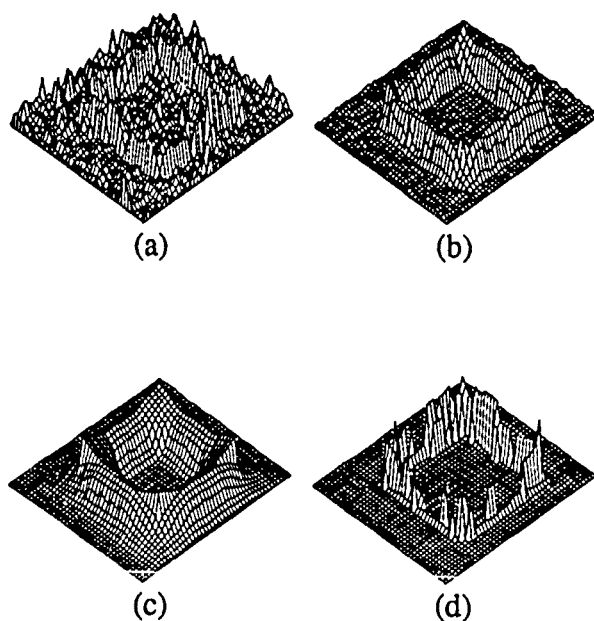
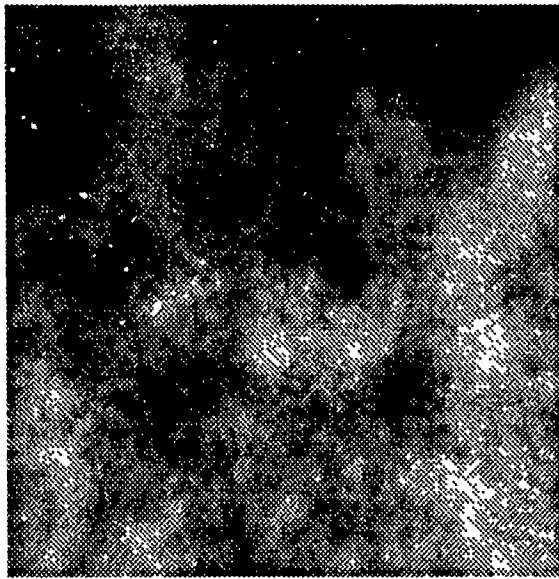


Figure 7: Isometric plots of the computed disparity by: (a) a 3×3 window; (b) a 7×7 window; (c) a 15×15 window; (d) the adaptive window algorithm.

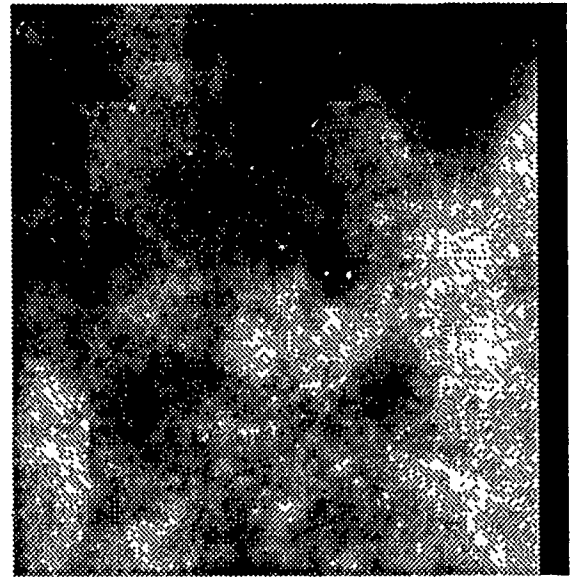


Window	Mean Error Value
3x3	0.22
7x7	0.20
15x15	0.34
Adaptive Window	0.08 (pixel)

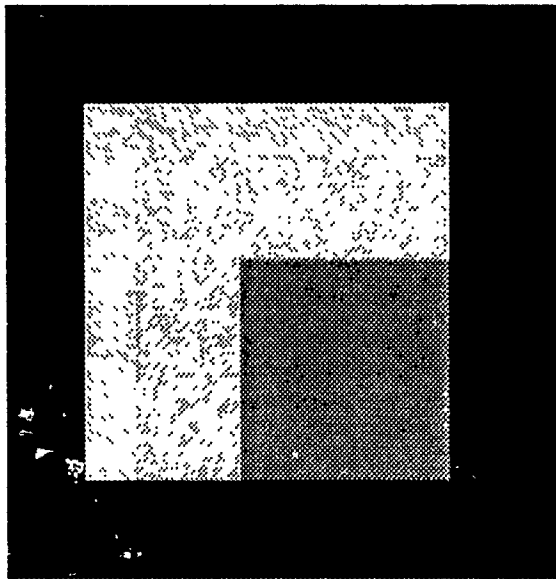
Figure 8: Difference between the true disparity and the computed disparity: (a) by a 3×3 window; (b) by a 7×7 window; (c) by a 15×15 window; (d) by the adaptive window.



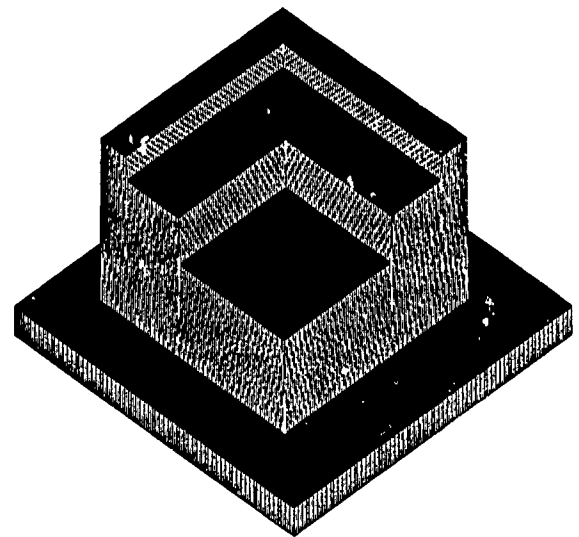
(a)



(b)

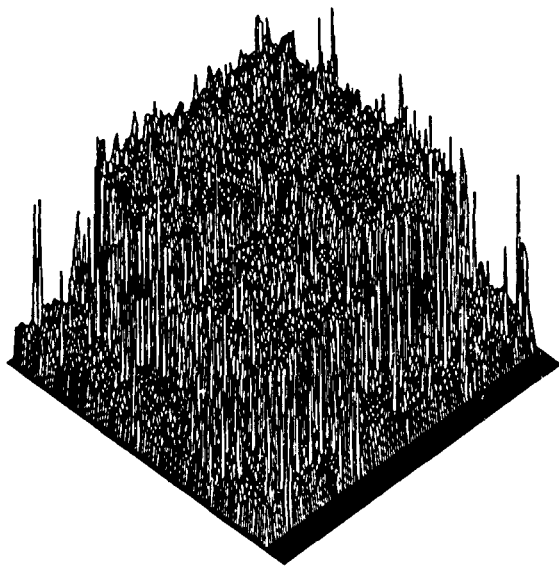


(c)

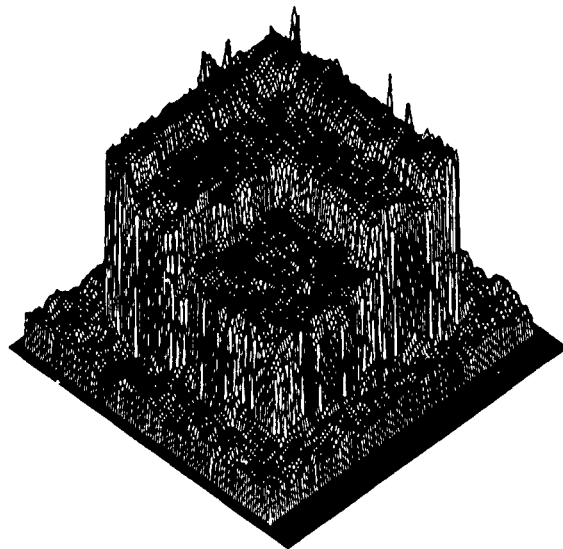


(d)

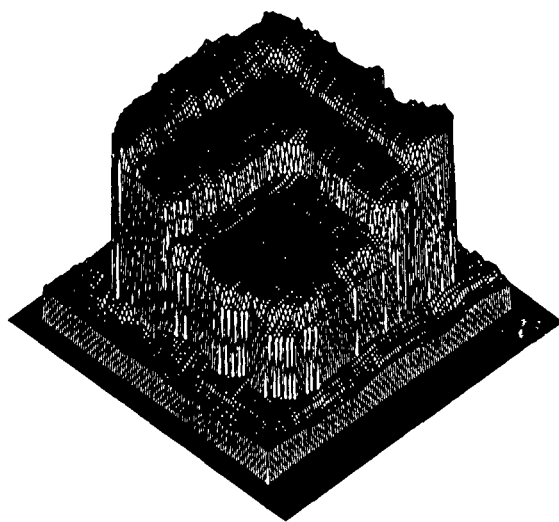
Figure 9: Synthesized stereo images no. 2: (a) Left image; (b) Right image; (c) Disparity pattern; (d) Isometric plot of the disparity pattern shown in (c).



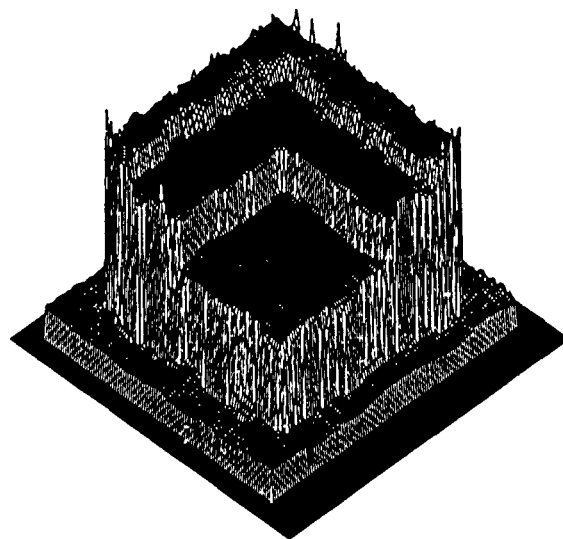
(a)



(b)



(c)

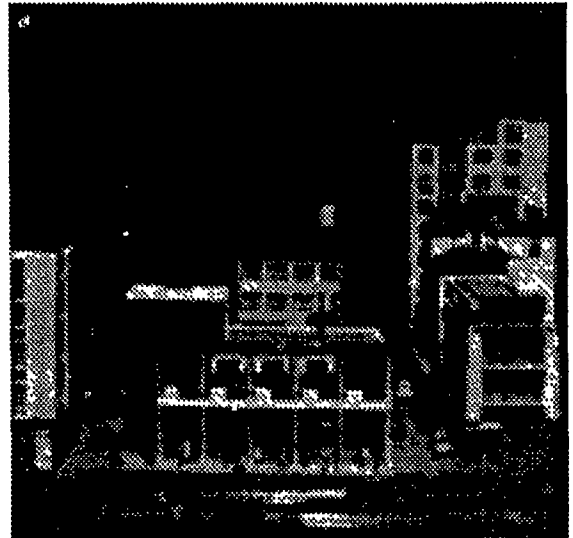


(d)

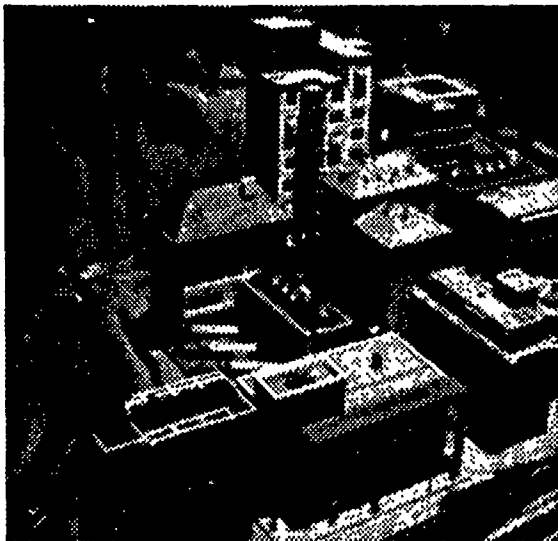
Figure 10: Computed disparities by: (a) a fixed 3×3 window; (b) a fixed 7×7 window; (c) a fixed 15×15 window; (d) the adaptive window.



(a)

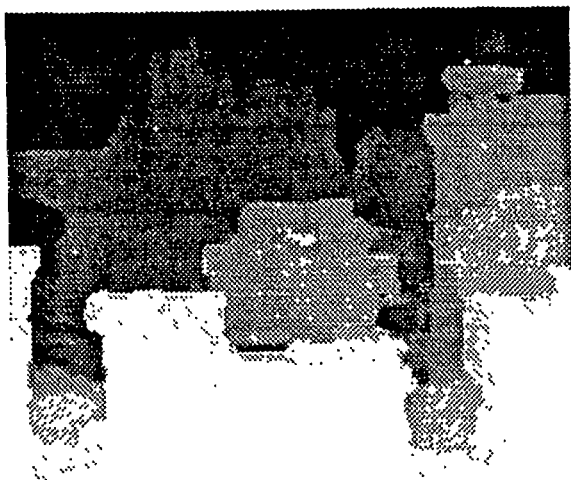


(b)

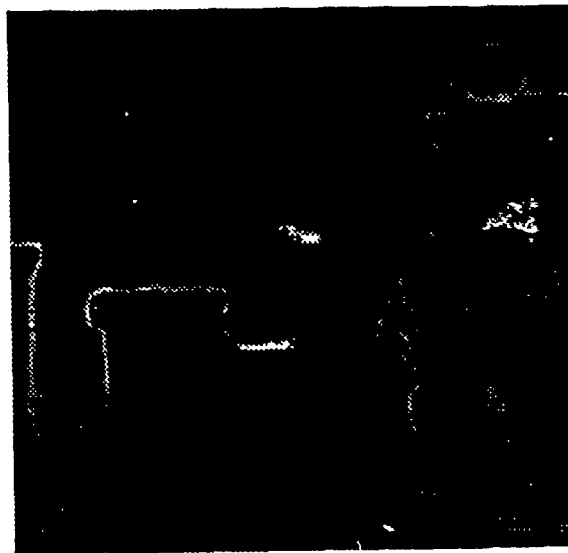


(c)

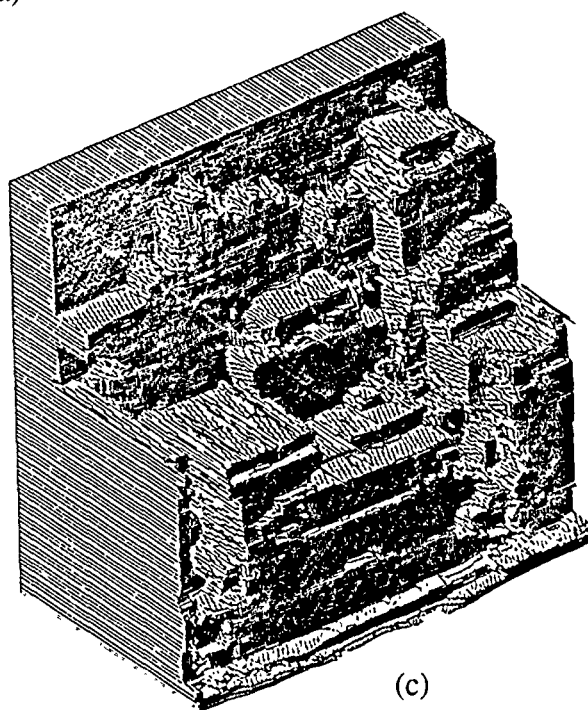
Figure 11: "Town" stereo data set: (a) Upper image of stereo; (b) Lower image of stereo; (c) Oblique view.



(a)

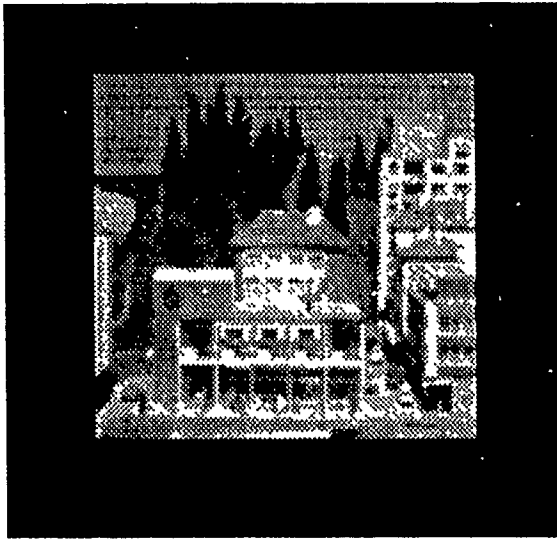


(b)

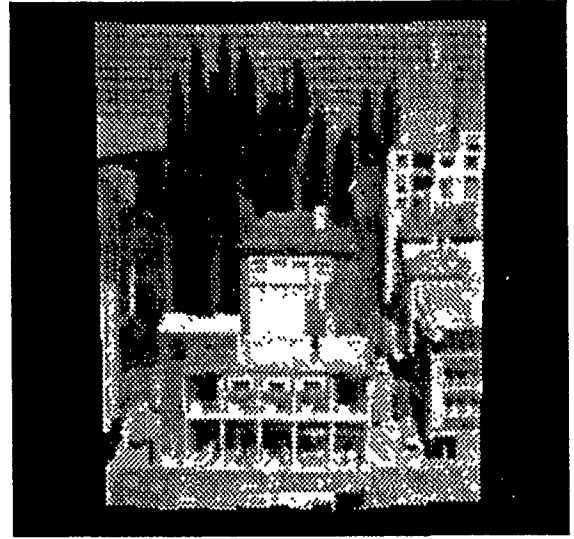


(c)

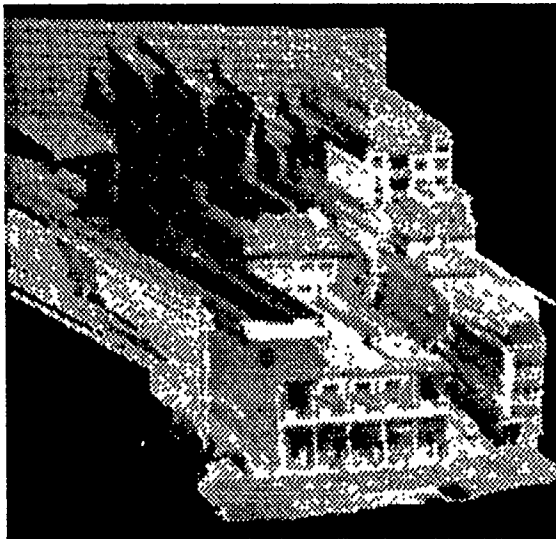
Figure 12: Computed disparity and uncertainty for the "town" stereo data: (a) Disparity map, (b) Uncertainty, (c) Isometric plot of the disparity map.



(a)



(b)

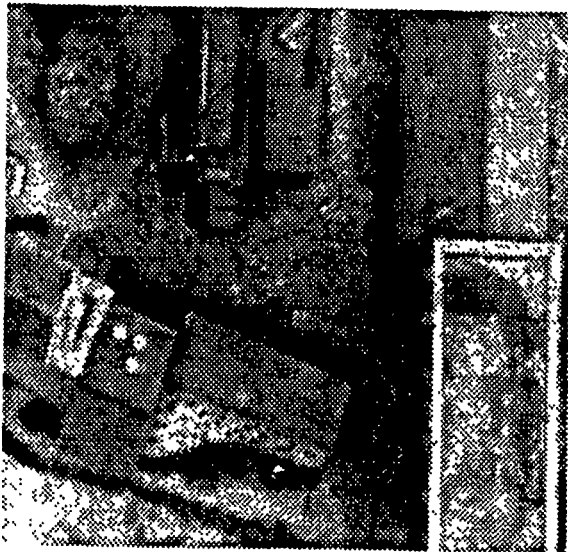


(c)

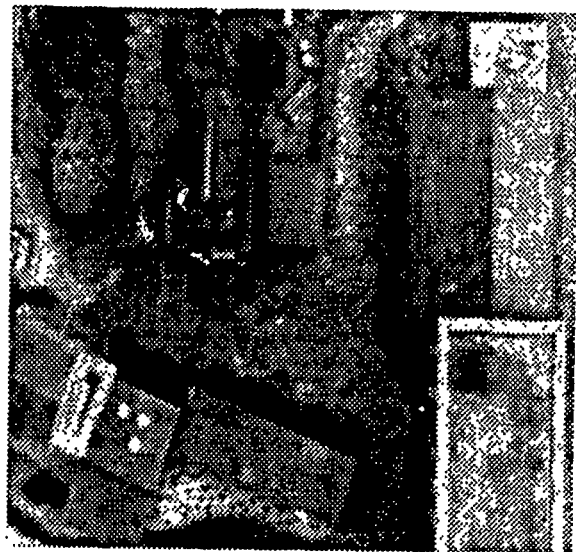


(d)

Figure 13. Perspective views of the recovered scene. (a) from the original camera position, (b) from an upper position; (c) from an upper left position; (d) from an upper right position.

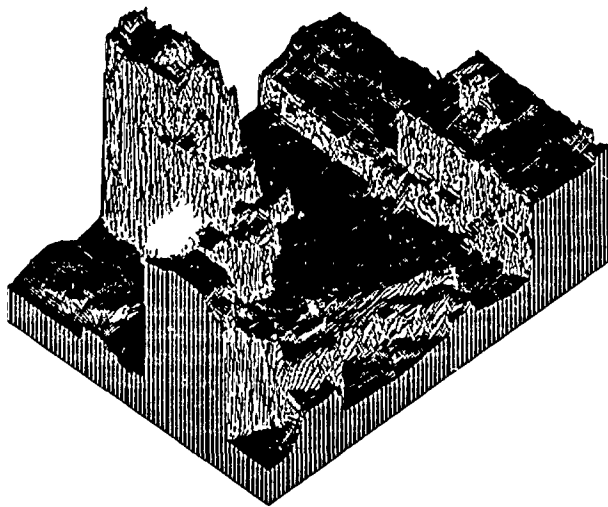


(a)

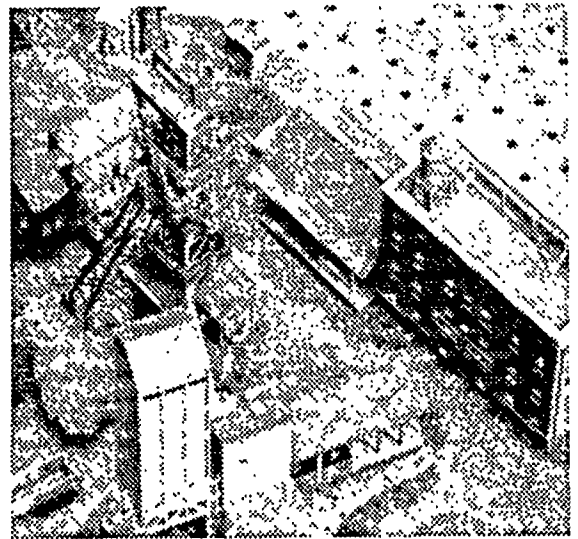


(b)

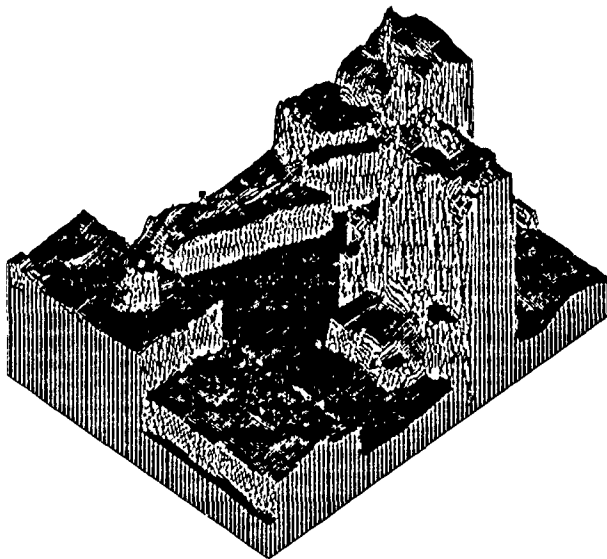
Figure 14: "Coal mine" stereo data set: (a) Lower image; (b) Upper image.



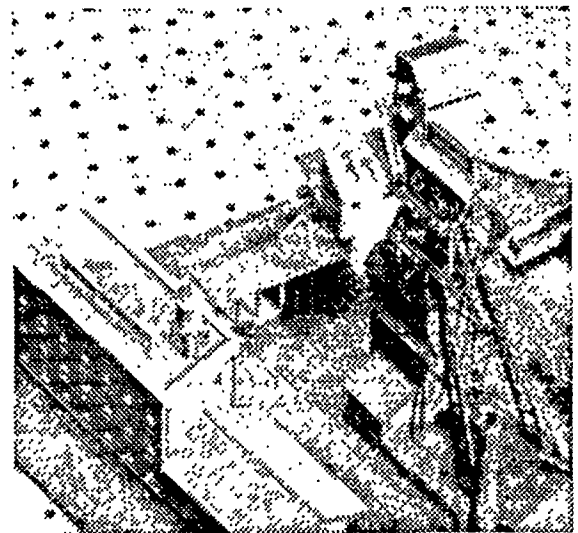
(a)



(b)



(c)



(d)

Figure 15: Isometric plots of the computed disparity map and their corresponding actual view: (a) (b) Isometric plot and corresponding view from the lower left corner; (c) (d) Isometric plot and corresponding view from the upper right corner.

A Approximating Distribution of $n_s(\xi, \eta)$

We will examine the statistical properties of $n_s(\xi, \eta)$, which is the right hand side of equation (4)

$$n_s(\xi, \eta) = (d_r(\xi, \eta) - d_r(0, 0)) \frac{\partial}{\partial \xi} f_2(\xi + d_r(0, 0), \eta) + n(\xi, \eta).$$

We see that $n_s(\xi, \eta)$ has the form of

$$xy + n,$$

where

$$\begin{aligned} x &= d_r(\xi, \eta) - d_r(0, 0), \\ y &= \frac{\partial}{\partial \xi} f_2(\xi + d_r(0, 0), \eta), \\ n &= n(\xi, \eta) \end{aligned}$$

Our assumptions are that: x is a zero-mean Gaussian noise; y and n are both zero-mean Gaussian white noise; and x , y , and n are statistically independent.

Let $p_x(x)$, σ_x^2 , and $R_x(\tau)$ denote the density function, variance, and autocorrelation function of x , respectively.

$$p_x(x) = \frac{1}{\sqrt{2\pi}\sigma_x} e^{-\frac{x^2}{2\sigma_x^2}}$$

We define notations for y and n in the same manner. Since y is white, we have

$$R_y(\tau) = a\delta(\tau),$$

where, $\delta(\tau)$ is the delta function and a is a constant.

First, let us examine the properties of random variable z which is product of x and y

$$z = xy.$$

Since x and y are independent, the autocorrelation function of z is given by (see [de Coulon, 1986]):

$$R_z(\tau) = R_x(\tau)R_y(\tau) = b\delta(\tau)$$

where b is a constant. Therefore, z is also white.

The density function $p_z(z)$ can be calculated as

$$\begin{aligned} p_z(z) &= \int_{-\infty}^{\infty} \frac{1}{|x|} p_x(x) p_y\left(\frac{z}{x}\right) dx \\ &= \frac{1}{\pi\sigma_x\sigma_y} \int_0^{\infty} \frac{1}{x} \exp\left(-\frac{x^2}{2\sigma_x^2} - \frac{z^2}{2\sigma_y^2 x^2}\right) dx \\ &= \frac{1}{\pi\sigma_x\sigma_y} K_0\left(\frac{|z|}{\sigma_x\sigma_y}\right), \end{aligned}$$

where $K_0(z)$ is the modified Bessel function of order 0

$$K_0(z) = \frac{1}{2} \int_0^{\infty} x^{-1} \exp\left(-x - \frac{z^2}{4x}\right) dx.$$

The thick curve in Figure 16 shows this density function. $p_z(z)$ is a monomodal distribution which is symmetrical about the mode at $z = 0$. For simplicity, it is reasonable to approximate the distribution by a Gaussian distribution that has the same mean and variance as $p_z(z)$, which are

$$\begin{aligned} E[z] &= E[x]E[y] \\ &= 0 \\ E[(z - E[z])^2] &= E[(xy)^2] = E[x^2]E[y^2] \\ &= \sigma_x^2\sigma_y^2 \end{aligned}$$

The faint curve in figure 16 shows the zero-mean Gaussian distribution $N(0, \sigma_x^2\sigma_y^2)$.

Once we approximate the white noise $z = xy$ by a Gaussian distribution, it is straightforward to see that $z + n$ is also a Gaussian white noise and to calculate its mean and variance, since it is sum of two Gaussian white noises. Hence, equation (11).

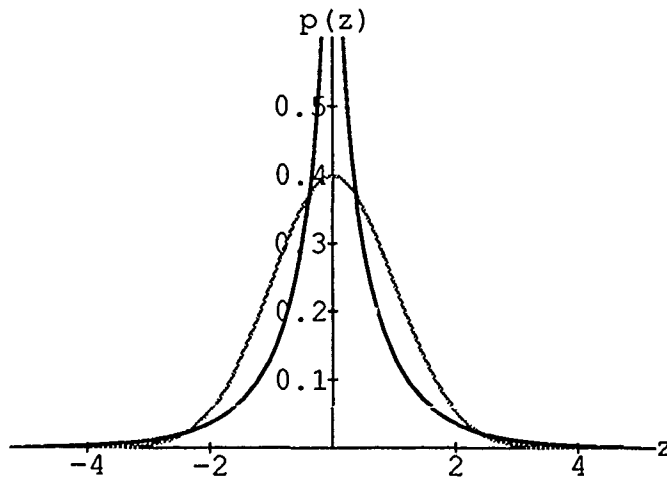


Figure 16: Probability density functions, $\frac{1}{\pi\sigma_x\sigma_y} K_0\left(\frac{|z|}{\sigma_x\sigma_y}\right)$ and $N(0, \sigma_x^2\sigma_y^2)$. The horizontal axis is normalized; i.e., $z' = \frac{z}{\sigma_x\sigma_y}$

General Model-Based 3D Surface Estimation, Recognition and Segmentation from Multiple Images*

David B. Cooper, Yi-Ping Hung, Jayashree Subrahmonia

Laboratory for Engineering Man/Machine Systems
Division of Engineering,
Brown University,
Providence, RI 02912

1 Introduction

A unified geometric-probabilistic approach is presented to 3D surface estimation, recognition and segmentation from two or more images. The central idea here is to model the joint probability of the images involved given the apriori unknown parameters in the model used. Then standard techniques from statistics and image processing can be used for parameter estimation, image segmentation and model recognition. Using this formulation, we illustrate algorithms for: estimating highly variable 3D surfaces, including depth discontinuities; estimating parameters for modeled curved surface; segmenting images into regions, each viewing a different 3D surface model; and recognizing objects viewed in the images. Other results not presented here include the Cramer-Rao lower bound on the error in surface reconstruction from the raw image data [Cernuschi-Frias *et al.*, 1989]. The situations that can be handled include virtually all those of interest in the stereo problem, and the results should be of maximum accuracy since the algorithms are either maximum likelihood or maximum a posteriori probability estimation or minimum probability of error recognition.

2 Problems of Interest and Basic Equations

Among the problems of interest are the following.

1. Maximum a posteriori probability (MAP) estimation of a 3D parameterized surface from a sequence of images.
2. Joint MAP recognition of 3D surface type and surface parameter estimation.
3. Minimum probability of error recognition of 3D surface type.

It turns out that for these objectives, it is also necessary to estimate the parameters specifying the pattern (i.e., reflectance, seen on the object surface). Consider a sequence of images $I_1(\cdot), I_2(\cdot), \dots, I_N(\cdot)$. Let α denote the vector of parameters for the object surface model, and let α be the vector of parameters that includes the surface parameters and the parameters that specify the

pattern seen on the surface. Then if l denotes the label for the object surface model, the solution to the first problem is

$$\max_{\alpha} p(I_1, \dots, I_N, \alpha) \propto \max_{\alpha} p(I_1, \dots, I_N | \alpha) p(\alpha) \quad (1)$$

The solution to the second problem is

$$\max_{l, \alpha} p(I_1, \dots, I_N | l, \alpha) p(\alpha | l) p(l) \quad (2)$$

The solution to the third problem is

$$\max_l \int_{-\infty}^{\infty} p(I_1, \dots, I_N | l, \alpha) p(\alpha | l) p(l) d\alpha \quad (3)$$

Note that in (2), $\max_{\alpha} p(I_1, \dots, I_N | l, \alpha)$ can be appreciably larger when l specifies *cylinder* than when l specifies *plane*, since a cylinder can always approximate the image data better than can a plane. Hence the presence of $p(\alpha | l)$ in equations (2) and (3) is usually necessary when the different surface types have different functional forms because, e.g., if a *plane* is present in the data, $p(\alpha | l)$ should be larger for $l = \text{plane}$ than for $l = \text{cylinder}$. The solution to (3) will have smaller recognition of surface type error than will the solution to (2), though, in many cases, the difference will not be significant.

More generally, there will be many objects in a scene. Assume the number of objects in a scene can vary. Then the generalization of the preceding is that what must be estimated is the number of surfaces present, their types, and their parameter values. These generalizations are the following.

4. MAP estimation of the number of surfaces present, the surface types, and their parameter values.
5. Minimum probability of error recognition of the number of surfaces present and their types.

Suppose there are K surfaces in the scene. Let $L^t = (l_1, l_2, \dots, l_K)$ denote the labels for these K surfaces, and $\alpha^t = (\alpha_1^t, \alpha_2^t, \dots, \alpha_K^t)$ denote the associated parameter vectors. Then the solution to problem 4 is

$$\max_{K, L, \alpha} p(I_1, \dots, I_N | \alpha, L) p(\alpha, L, K) \quad (4)$$

Note that the solution to (4) is also a minimum description length (MDL) solution [Leclerc, 1989]. Finally, the solution to problem 5 is

$$\max_{K, L} \int_{-\infty}^{\infty} p(I_1, \dots, I_N | \alpha, L) p(\alpha, L, K) d\alpha \quad (5)$$

*This work was partially supported by NSF Grant #IRI-8715774 and NSF-DARPA Grant #IRI-8905436

Equations (1)-(5) are reasonable to compute. We now briefly discuss the functions involved.

There are a number of models possible for I_n . In [Cooper *et al.*, 1988], a polynomial contour model is used. In the present paper, we assume $I_n(u)$ is some true noiseless image $\mu_n(u)$ plus white gaussian noise having mean 0 and variance σ^2 . We assume a pinhole camera model and a Lambertian surface for the object. Then a point P on the object surface is seen with equal brightness in all the images. Hence, the function in (1) can be written approximately as

$$\prod_{n=1}^N (2\pi\sigma^2)^{-d/2} \exp\left\{-\frac{1}{2\sigma^2} \sum_{u \in D_n} [I_n(u) - \mu_n(u)]^2\right\} \quad (6)$$

where D_n is the set of pixels in the n th image, and d is the number of pixels in D_n . The $\mu_n(u)$ are functionally related. This can be seen as follows. Consider Figure 1 illustrating the geometry for the two images, I_1 and I_2 .

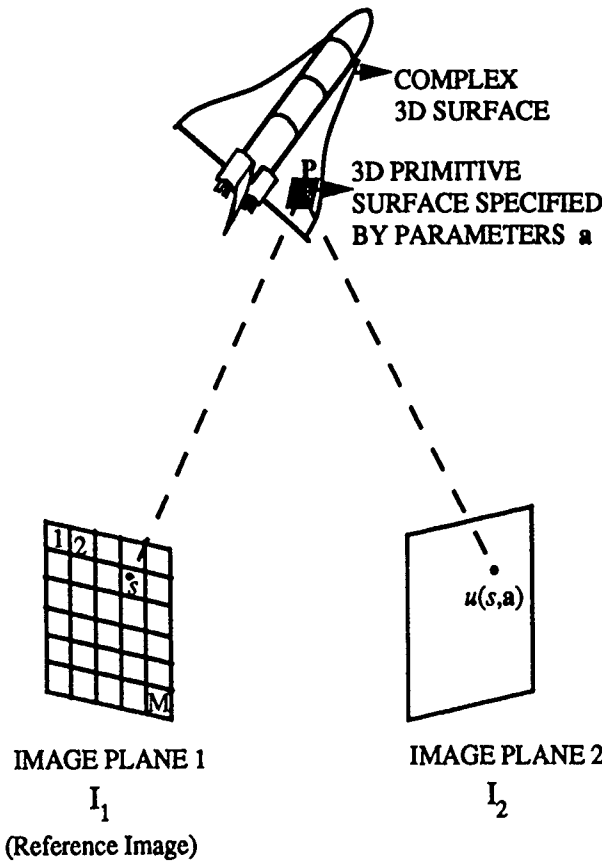


FIGURE 1

Take any point in image 1. Since the two cameras are calibrated, it is known that the surface point P seen at s lies along the backward projected line from s through the camera lens center. Assume the surface point seen lies on the surface specified by the parameter vector a . Then the point P seen is the intersection of the backward

projected ray determined by s with the surface specified by a . This point is seen in image 2 at the intersection of the image plane at camera 2 with the forward projected ray from P , through the lens center of camera 2. Hence, we denote this point in I_2 by $u(s, a)$. Because of the Lambertian assumption, if a is the true parameter value, we have

$$\mu_n(u(s, a)) = \mu_1(s) \quad (7)$$

Consequently, an approximation to (6) is

$$\prod_{n=1}^N (2\pi\sigma^2)^{-d/2} \exp\left\{-\frac{1}{2\sigma^2} \sum_{s \in D_1} [I_n(u(s, a)) - \mu_1(s)]^2\right\} \quad (8)$$

Unfortunately, the values $\mu_1(s)$ are apriori unknown. Hence, they must be treated as apriori unknown parameters. They specify the pattern on the 3D surface. The apriori unknown parameters α then consist of the 3D surface parameters a , the pattern parameter vector μ_1 having components $\mu_1(s)$, $s \in D_1$, and perhaps σ^2 . When $N=2$, equation (8) has some very nice properties. In particular, if the $\mu_1(s)$ are independent, uniformly distributed random variables, then

$$\max_{\mu_1} p(I_1, I_2 | a, \mu_1) p(\mu_1) p(a) = (2\pi\sigma^2)^{-d/2} \exp\left\{-\frac{1}{4\sigma^2} \sum_{s \in D_1} [I_1(s) - I_2(u(s, a))]^2\right\} \times \quad (9)$$

$$p(\mu_{1MAP}) p(a)$$

where $\mu_{1MAP}(s) = \frac{1}{2}[I_1(s) + I_2(u(s, a))]$ is the MAP estimate of $\mu_1(s)$, and $p(\mu_{1MAP})$ is constant over a rectangular solid. For $3 \leq N$, the situation is more complex, but can be treated effectively. See [Hung *et al.*, 1990]. Equation (9) is to be used in the maximization (2) when $p(\mu_1)$ is the uniform probability density function and $N=2$.

One other equation that facilitates the practical computation of equations (1)-(6) is

$$\frac{p(I_1, \dots, I_N | \alpha)}{p(I_1, \dots, I_N | \hat{\alpha}_N)} \approx \exp\left\{-\frac{1}{2}(\alpha - \hat{\alpha}_N) \Phi_N (\alpha - \hat{\alpha}_N)\right\} \quad (10)$$

where $\hat{\alpha}_N$ is the maximum likelihood estimate (MLE) of α based on the image data I_1, \dots, I_N , and Φ_N is the 2nd derivative matrix having i, j th component [Hung *et al.*, 1990]

$$-\frac{\partial^2}{\partial \alpha(j) \partial \alpha(i)} \ln p(I_1, \dots, I_N | \hat{\alpha}_N)$$

Hence all the useful information about α is summarized in the quadratic form. When (10) is used for $p(I_1, \dots, I_N | l, \alpha)$ in (3), the result is

$$\max_l p(I_1, \dots, I_N | l, \hat{\alpha}_N) \times (2\pi\sigma^2)^{-q/2} |\Phi_N|^{-1/2} p(\hat{\alpha}_N | l) p(l)$$

where q is the number of components in α , and $|\Phi_N|$ is the determinant of Φ_N . Note that this should give better recognition results than use of (2), because, e.g., $|\Phi_N|^{-1/2}$ will in general be smaller for a cylinder than for a plane.

We now discuss how problems 1-5 can be solved and show examples of experiments for the solutions to two problems.

3 A Computationally Practical Approach to Surface Estimation

In order to use parallel processing to achieve real time operation, we partition an image, e.g., the first, into square windows, assume each window is a view of a single 3D surface and first estimate these surfaces independently in parallel. (Within each window, parallel processing can also be used.) Then, these initial estimates can be used as a starting point for global estimation of the entire complex surface seen in the images. Since gradient descent is the optimization technique used in most of our solutions, some care must be exercised to deal with the multimodal functions involved. We handle this at the individual window estimation stage, as discussed in [Cernuschi-Frias *et al.*, 1989, Hung *et al.*, 1990].

4 Sequential Algorithm for Surface Estimation

Here we want to process each image as it is received to extract what is useful, and then discard it, thus not having to store all the images or to process them simultaneously. The final estimate based on N images should be roughly as if all the images were processed at the same time. We do this by making use of (10).

$$\begin{aligned} p(I_1, \dots, I_{N+1} | \alpha) \\ = p(I_1, \dots, I_N | \alpha) p(I_{N+1} | \alpha) \\ \approx p(I_1, \dots, I_N | \hat{\alpha}_N) \times \\ \exp\{-\frac{1}{2}(\alpha - \hat{\alpha}_N) \Phi_N(\alpha - \hat{\alpha}_N)\} p(I_{N+1} | \alpha) \end{aligned} \quad (11)$$

Then $\hat{\alpha}_{N+1}$ is obtained by maximizing (11) with respect to α , and Φ_{N+1} is obtained from (11) by $\Phi_{N+1} = \Phi_N + \Delta_{N+1}$ where Δ_{N+1} is a matrix having i, j th element

$$-\frac{\partial^2}{\partial \alpha(j) \partial \alpha(i)} \ln p(I_{N+1} | \hat{\alpha}_{N+1}) \quad (12)$$

Note that this algorithm makes full use of the multiple images and the increasing camera baseline, and the resulting expressions can also be used for object recognition [Hung *et al.*, 1990, Hung, 1989].

5 MAP Estimation of the Number of Surfaces, Their Types and Their Parameter Values

The solution to problem 4 is given in equation (4). For two images this equation becomes

$$\max_{K, L, \alpha} p(I_1, I_2 | \alpha, L) p(\alpha, L, K) \quad (13)$$

Suppose the reference image I_1 is partitioned into M windows, and t^m denotes the label associated with the surface patch seen in the m^{th} window, i.e. $t^m \in L$. Then, the problem can be equivalently stated as getting the MAP estimate of $K, T = (t^1, \dots, t^M)$ and α , i.e., finding the K, T and α that maximize the following posterior probability

$$p(I_1, I_2 | \alpha, T) p(\alpha, T, K) \quad (14)$$

$p(\alpha, T, K) = p(\alpha | T) p(T | K) p(K)$. We take $p(K)$ to be uniformly distributed over some interval and model

$T | K$ with a Markov Random Field (MRF) by choosing an appropriate neighbourhood system and its associated potential functions on all the cliques. Unless prior information is available, we take

$$p(\alpha | T) = \prod_{k=1}^K p(\alpha_k | l_k)$$

, and α_k is uniformly distributed over some region. Note, we assume the $p(\alpha_k | l_k)$ are independent of one another, and that all $p(\alpha_k | l_k)$ for the same type of surface, e.g., a sphere, are the same. Using the MRF-Gibbs equivalence [Besag, 1974], the prior distribution for $T | K$ can be written as follows.

$$p(T | K) = \frac{1}{Z} \exp\{-U(T)\} \quad (15)$$

where Z is a constant and the energy function is of the form

$$U(T) = \sum_{c \in C} V_c(T) \quad (16)$$

with $V_c(T)$ being the contribution of the clique c to the energy of T . Here, C denotes the set of all cliques with respect to the chosen neighbourhood system. In this paper, the first order neighbourhood system is used, where the cliques are individual sites and pairs of adjacent horizontal and vertical sites, and the potential for a clique $c = \{m, j\}$ is chosen to be:

$$V_{\{m, j\}}(T) = V_{\{m, j\}}(t^m, t^j) = \begin{cases} 0 & \text{if } t^m = t^j \\ \beta/4 & \text{if } t^m \neq t^j \end{cases} \quad (17)$$

The potential functions for the cliques is not restricted in any way, and can be designed to formulate a wide variety of fields. The preceding MRF is homogeneous. It tends to generate blob-like regions. Blob size increases with increasing β . It is the simplest field, and was designed solely to discourage segmentation into small regions or regions with wiggly boundaries.

We now focus on the data term in Equation (14), $p(I_1, I_2 | \mu_{1MAP}, \mathbf{a}, T)$. Let I_1^m denote the image intensity seen in the m^{th} window in image 1 and I_2^m be its corresponding image data seen in image 2. Let \mathbf{a}^m denote the parameter vector that describes the 3D surface patch seen in the m^{th} window. Then, \mathbf{a}^m is the surface parameter vector associated with the surface label t^m . Since the noise in an image is white and independent of the noise in the other image,

$$p(I_1, I_2 | \mu_{1MAP}, \mathbf{a}, T) = \prod_{m=1}^M p(I_1^m, I_2^m | \mu_{1MAP}^m, t^m, \mathbf{a}) \quad (18)$$

which, from section 2 is

$$\begin{aligned} p(I_1, I_2 | \mu_{1MAP}^m, \mathbf{a}, T) \propto \\ \prod_{m=1}^M \exp\{-\frac{1}{4\sigma^2} \sum_{s \in D^m} [I_1(s) - I_2(\mathbf{u}(s, \mathbf{a}^m))]^2\} \end{aligned} \quad (19)$$

Then, the MAP estimation of the number of surfaces present, the surface types and their parameter values becomes the maximization of

$$p(I_1, I_2 | \mu_{1MAP}, \mathbf{a}, T) p(\mathbf{a}) p(T | K) p(K) \quad (20)$$

Stochastic relaxation [Geman and Geman, 1984] for maximizing (20) is computationally too costly. Deterministic relaxation, such as the Iterative Conditional Means (ICM) [Cohen *et al.*, 1984, Besag, 1986] is computationally simple but is only guaranteed to find a local maximum. It is necessary to have fairly decent initial estimates for a , T and K to get good final estimates. Therefore, the first two steps of the algorithm are computationally reasonable operations for getting the initial estimates for a , T and K .

To start out, the reference image is partitioned into M_0 windows. A maximum likelihood estimate (m.l.e) for the 3D surface patch seen in each window is computed. The estimate for a window is carried out independently of those for other windows. Here, the window size is chosen to be small enough that the 3D surface patches viewed in most windows are of a single surface, but large enough that there are enough data present in most windows for acquiring good local surface estimates.

Once the maximum likelihood estimates for the 3D surface seen in each window is obtained, any clustering algorithm can be used to decide the number of surfaces present and the parameters for each surface. In the experiments run with a limited data set, the K-means algorithm with a simple performance functional provided very satisfactory results. The inputs to the K-means algorithm are the m.l.e's for the 3D surface patches seen in all the windows.

The experimental results using real images are shown in Figures 2-5. Experiments are run on two different data sets. All the experiments are run using only planes as primitives. The maximum likelihood surface estimation has been implemented for a larger primitive set consisting of planes, cylinders and spheres and we are in the process of implementing the MAP segmentation and estimation for this larger primitive set. The first data set is a portion of a cube consisting of two planes. Figures 2(a) and 2(b) show the two images used of the cube. Figure 2(c) shows the 3D surface, the camera positions and the direction from which the reconstruction is viewed. In the experiment, patches of size 40×40 pixels were used to do the local surface patch estimation. Figure 3(a) shows the surface reconstruction after the local surface patch estimation. The surface reconstruction after the unsupervised K-means clustering is shown in Figure 3(b). We see that the unsupervised K-means clustering technique comes up with three classes. The two large classes are well fit by the model and the small one inbetween is not and is rejected by the performance functional (14). For the final MAP segmentation and estimation, patches of size 10×10 pixels are used. The surface reconstruction after the fine segmentation is shown in Figure 3(c). It is seen that the patches belonging to the reject class get classified into one of the other classes, thus producing two classes corresponding to the two planes.

For the second experiment shown, the scene consists of three cylinders (c1, c2 and c3) and two planes (p1 and p2). In this example, the two planes fit the primitive model well whereas the three cylinders cannot be well approximated by a single primitive planar patch each. The purposes of this experiment were first to see

whether planar surfaces are found when nonplanar surfaces are present, and second to explore the quality of the planar approximation to nonplanar surfaces. Figures 4(a) and 4(b) show the two images used as input. Two of the cylinders and one of the planes have shiny surfaces. Patches of size 32×32 pixels are used to do the local surface estimation and the unsupervised K-means clustering. The unsupervised K-means clustering program in this case comes up with 12 classes. Five of them have small likelihood of data and are therefore rejected by the performance functional (14). Two of the classes correspond to the two planes, p1 and p2 and are well fit by the model. The cylinders c1 and c3 are approximated by one plane each and cylinder c2 is approximated by three planes. The reason for this is that c1 and c3 have small curvatures and only a small portion of each of these cylinders is seen in both images. Cylinder c2, however, has large curvature and most of it is seen in both images. For the final MAP segmentation and reestimation using a MRF, windows of size 16×16 pixels were used. Figure 5(a) shows the segmentation of the portion of the reference image that is seen in both the input images. Here each region of constant intensity corresponds to one plane found. The surface reconstruction after the fine segmentation is shown in Figure 5(b).

The details of the algorithm are given in [Subrahmonia *et al.*, 1990, Chou, 1988]. We are exploring the limits to the smallest surfaces that can be found with this approach.

6 Estimation of Highly Variable Surfaces

Our approach to the estimation of highly variable surfaces is to model the surface as a stochastic process and then do MAP estimation. For this purpose, the vector a models the surface, and the surface estimation is given by (1). Perhaps, the simplest surface model is the depth map, and a^m is then the surface depth at the m^{th} pixel in image 1. Or, image 1 may be partitioned into M_0 windows, and the surface depth modeled as being constant over a window. Then a^m is this surface depth over the m^{th} window in image 1.

Figures 7(b) show the results of the Surface MAP Estimation based on the images shown in Figure 6. The images are of a cylinder lying on a planar surface. Figure 7a shows the depth map reconstructed based on the two images by using the maximum likelihood estimates computed independently for each window in image 1. While this reconstructed depth map reveals the structure of a cylindrical box lying on a flat surface, it is noisy because each depth parameter is estimated using intensity data in only a small window. By modeling both the smoothness and the discontinuities of the 3D surfaces with a doubly stochastic process, specifically a coupled Markov Random Field [Cohen and Cooper, 1987, Geman and Geman, 1984], we obtain the better reconstruction shown in Figure 7b [Hung and Cooper, 1990].



FIGURE 2(a)

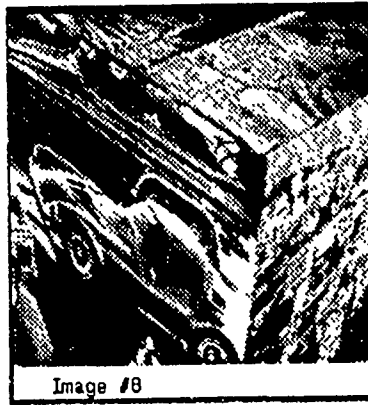


FIGURE 2(b)

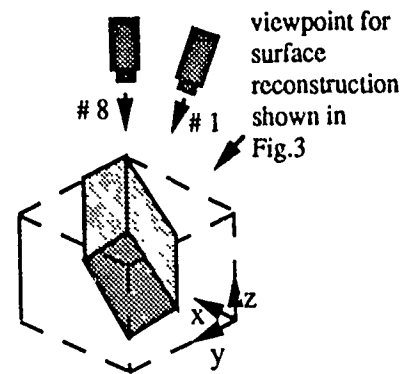


FIGURE 2(c)

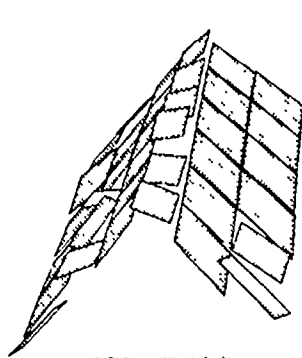


FIGURE 3(a)

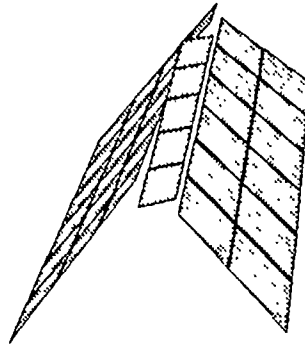


FIGURE 3(b)

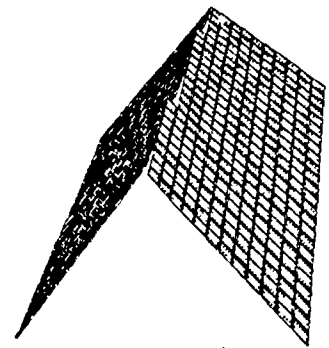


FIGURE 3(c)



FIGURE 4(a)



FIGURE 4(b)

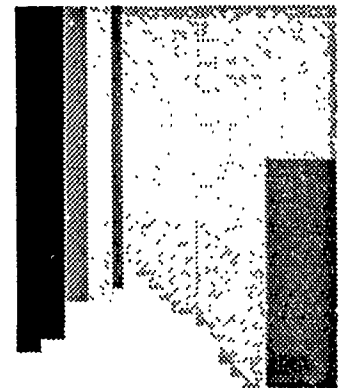


FIGURE 5(a)

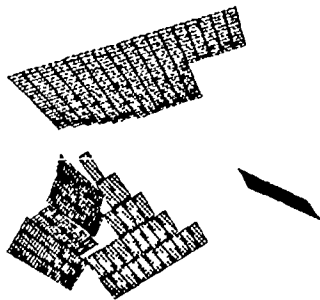


FIGURE 5(b)

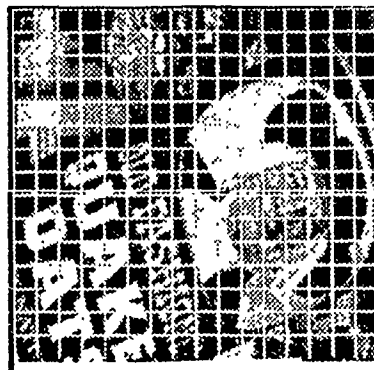


FIGURE 6(a)



FIGURE 6(b)

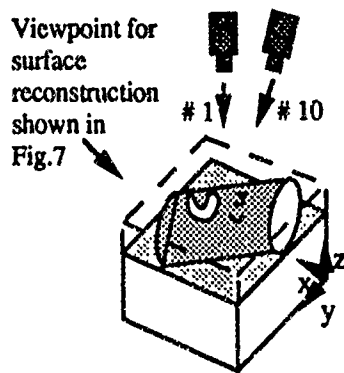


FIGURE 6(c)

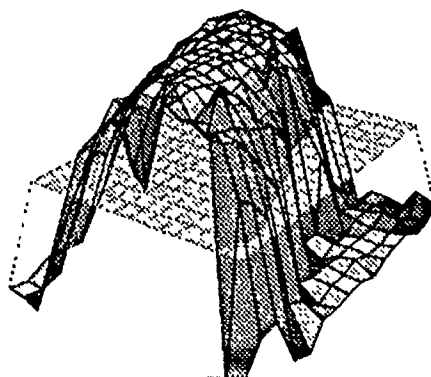


FIGURE 7(a)

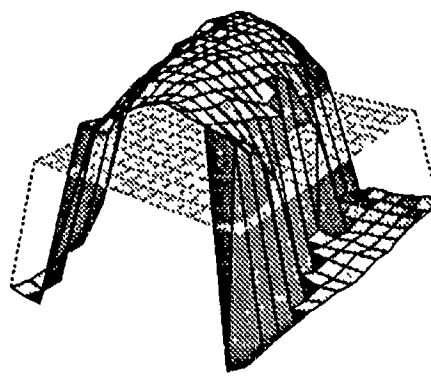


FIGURE 7(b)

References

- [Besag, 1974] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of Royal Statistical Society, B* 36:192-236, 1974.
- [Besag, 1986] J. Besag. On the statistical analysis of dirty pictures. *Journal of Royal Statistical Society, B* 48:259-302, 1986.
- [Cernuschi-Frias et al., 1989] B. Cernuschi-Frias, D.B. Cooper, Y.P. Hung, and P.N. Belhumeur. Toward a model-based Bayesian theory for estimating and recognizing parameterized 3D objects using two or more images taken from different positions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1028-1052, October 1989.
- [Chou, 1988] P.B. Chou. The theory and practice of Bayesian image labelling. Technical Report 258, Computer Science Dept., University of Rochester, August 1988.
- [Cohen and Cooper, 1987] F. Cohen and D.B. Cooper. Simple parallel hierarchical and relaxation algorithms for segmenting non-causal Markovian random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 195-219, March 1987.
- [Cohen et al., 1984] F. Cohen, D.B. Cooper, J.F. Silverman, and E.B. Hinkle. Simple parallel hierarchical and relaxation algorithms for segmenting textured images based on noncausal Markovian random field models. In *Proceedings, IEEE Seventh International Conference on Pattern Recognition*, pages 1104-1107, July 1984.
- [Cooper et al., 1988] D.B. Cooper, Y.P. Hung, and G. Taubin. A new model-based stereo approach for 3D surface reconstruction using contours on the surface pattern. In *Proceedings, IEEE International Conference on Computer Vision*, pages 74-83, December 1988.
- [Geman and Geman, 1984] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 721-741, November 1984.
- [Hung and Cooper, 1990] Y.P. Hung and D.B. Cooper. Maximum a posteriori probability 3D surface reconstruction using multiple intensity images directly. In *Proceedings, SPIE Conference on Sensing and Reconstruction of 3D Objects and Scenes*, February 1990.
- [Hung et al., 1990] Y.P. Hung, D.B. Cooper, and B. Cernuschi-Frias. Asymptotic Bayesian surface estimation using an image sequence. Technical Report LEMS-73, Brown University, June 1990.
- [Hung, 1989] Y.P. Hung. *Three Dimensional Surface Reconstruction using a Moving Camera: A Model-Based Probabilistic Approach*. PhD thesis, Brown University, November 1989.
- [Leclerc, 1989] Y.G. Leclerc. Image and boundary segmentation via minimum-length encoding on the connection machine. In *Proceedings, DARPA Image Understanding Workshop*, pages 1056-1069, May 1989.
- [Subrahmonia et al., 1990] J. Subrahmonia, Y.P. Hung, and D.B. Cooper. Model-based segmentation and estimation of 3D surfaces from two or more intensity images using Markov random fields. In *Proceedings, Tenth International Conference on Pattern Recognition*, pages 390-397, June 1990.

FLAT MMF: A New Recursive Technique for the 3-D Motion Analysis^{*}

Siu-Leong Iu

Kwangyoen Wohn

The Grasp Laboratory
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104.

Abstract

Previous attempts to the 3-D motion interpretation which utilize a large number of frames require a priori model that limits their usage to the motion allowed by the model only. In real environment, however, 3-D motion is seldom described by the simple model such as constant angular velocity, polynomial trajectories.

In this paper, we identify three sources of model mismatch - parameter jumping, undermodeling and overmodeling - and analyze the performance degradation due to each of them. We find that these model mismatches cause a very large estimation error or even make the estimation process diverge sometimes. As a solution, we propose a new recursive filter called the *Finite Lifetime Alternately Triggered Multiple Model Filter* (FLAT MMF). A number of simulations are conducted to illustrate the performance degradation of the conventional filter due to the model mismatches and the performance improvement when the proposed FLAT MMF is used.

1. INTRODUCTION

The problem of estimating the 3-D motion and structure of moving objects in space from time-varying images has attracted many vision researchers for more than ten years. Many interesting results on the theoretical and computational aspects of the problem have been published, but there is no canonical algorithm accepted by the majority of researchers, that can be applied to a wide range of time-varying images. This is partly due to the fact that each individual algorithm counts on the different type of image features or on the different assumptions about the motion and the scene. Perhaps another, bigger reason is that most of previous analyses rely on a small number of image frames (typically two or three) to derive the constraint between the image and the 3-D motion, and as a result, the algorithms suffer from the serious instability with respect to the input noise.

Recently, a number of researchers have proposed to use a large number of frames in the attempt to beat down the effect of noise [Weng87, Broi86a,b, Boll85, Iu89, Kuma89]. The 3-D motion estimate is smoothed not only spatially, but also temporally. The major challenge in the "extended" motion analysis is, unlike the "instantaneous" motion, that the temporal behavior of 3-D

motion during the extended time interval must be modeled properly so that it can be used to derive the motion constraint. Therefore, a common assumption of "rigidity of object" often used in the instantaneous analysis is not sufficient for the extended analysis. The causality of 3-D motion must be specified in an explicit form. [Boll85] assumes that the motion involves only translation. [Weng87] assumes that the translation is given as a polynomial while the external torque is zero. Broida and Chellapa published a series of articles in which both translation and rotation were modeled in terms of power series of known degrees [Broi86a,b]. In a similar vein, we have shown that, by imposing the temporal constraint, the 3-D motion of a single feature point could be recovered up to a scale factor [Iu89].

All of these extended-time analyses are based upon the assumption that the actual motion is matched to the a priori model which has to be fixed by the user prior to the execution of algorithm. An example would be a model which requires the object to move with a constant acceleration over the time interval of interest. Since an object may move almost arbitrarily in front of the camera, such model becomes invalid as the length of the observation time interval increases, although the model fits pretty well during a short initial period. Such model mismatch may cause a very large estimation error or may even make the estimation process diverge.

The above observation motivates us to study various problems associated with the model mismatch. For an object that moves rather freely in front of the camera, there are three classes of model mismatch we must consider. The first one is "parameter jumping" in which the 3-D motion parameters change abruptly at a finite number of instances. An example is a bouncing ball off the floor. In this case the trajectory of motion parameters is piecewise continuous. The second class of model mismatch is called "undermodeling" in which the computational model does not fully describe the actual motion, but is a good approximation within some interval. This case interests us most, since we will need to adopt a generic model which may not be fit to all types of motion, but may fit to a large bodies of natural motion that occur frequently in the real world. The last is "overmodeling" in which the actual motion is simpler than the model being used. This case is less serious than undermodeling, but as we shall see, the overmodeling may drop down the performance of algorithm.

In this paper, we shall first analyze the performance degradation due to the model mismatch, and propose a new parallel, recursive algorithm which handles all three classes of model mismatch. The new algorithm is called *Finite Lifetime Alternately Triggered Multiple Model Filter* (FLAT MMF). Our analytical derivation

^{*} This research was supported in part by DARPA grant N00014-88-K-0632 and NSF grant IR189-06770.

will be limited to the "2-D" motion since it is impossible to obtain a closed-form solution for the 3-D motion for the extended time interval. However, experiments were done on the 3-D motion as well as 2-D motion. The experimental result will be presented in Section 5.

Also we limit the scope of paper to the (2-D and 3-D) motion of a single feature point. This is necessary because of the following reasons: (1) To make mathematical formula in the paper compact by reducing the number of parameters involved, and (2) to conduct a formal analysis on the performance of FLAT MMF against the conventional recursive technique. In Section 2 we begin with our discussion on the motion of a single feature point in space.

2. MOTION OF A FEATURE POINT

In our earlier work [Lu89] we have shown that the 3-D motion of a single feature point could be recovered up to a scale factor. The motion of a point was approximated in terms of a power series of predetermined degree. A batch approach was used to solve the nonlinear optimization problem that tried to minimize the difference between the model and the observation. Unfortunately, the batch approach cannot handle the model mismatch we have discussed briefly in the previous section. Here, we reformulate the problem as a state estimation problem. Let $\underline{P}(t) = [X(t) \ Y(t) \ Z(t)]^T$, $\underline{p}(t) = [x(t) \ y(t)]^T$ and $\underline{V}(t) = [V_X(t) \ V_Y(t) \ V_Z(t)]^T$ be the 3-D position, the projected position and the instantaneous velocity of the moving particle at time t , respectively. The symbol "T" denotes the transpose of a vector. Then we have

$$\frac{d}{dt} \underline{P}(t) = \underline{V}(t), \quad (2.1)$$

$$x(t) = X(t) / Z(t), \quad y(t) = Y(t) / Z(t), \quad (2.2a,b)$$

under the perspective projection with the normalized focal length. If the particle moves smoothly in the 3-D space, we may model its trajectory as the following power series.

$$\underline{P}(t) = \sum_{n=0}^{d_m} \underline{P}^{(n)}(t_0) \frac{(t - t_0)^n}{n!}, \quad (2.3)$$

where d_m is the order of the model of the translation and $\underline{P}^{(n)}(t_0) = [X^{(n)}(t_0) \ Y^{(n)}(t_0) \ Z^{(n)}(t_0)]^T$ is the n -th derivative of $\underline{P}(t)$ evaluated at time $t = t_0$. Note that $\underline{P}^{(0)}(t_0) = \underline{P}_0(t_0)$. The physical interpretation of $\underline{P}^{(1)}(t_0)$ and $\underline{P}^{(2)}(t_0)$ are the velocity and the acceleration of translation at time t_0 , respectively. It is well known that we can only recover the translation up to a scalar factor because of the nature of projection. Then our goal is to estimate the translational coefficients $\underline{P}^{(n)}(t_0)$, $n = 0, 1, \dots, d_m$ in (2.3) scaled by a factor representing the absolute depth, given the measurements of $\underline{p}(t_j)$, $j = 0, 1, \dots$.

We may formulate the problem discussed above as the nonlinear state estimation problem. Let the state vector $\underline{s}(t)$ which is composed of the unknown parameters, and the measurement vector $\underline{m}(t)$ which is formed by the available measurements, satisfy the following plant and measurement equations,

$$\frac{d}{dt} \underline{s}(t) = \underline{f}(\underline{s}(t)), \quad (2.4)$$

$$\underline{m}(t_j) = \underline{h}(\underline{s}(t_j)) + \underline{n}(t_j), \quad (2.5)$$

where $\underline{f}(\cdot)$ and $\underline{h}(\cdot)$ are nonlinear vector functions, and $\underline{n}(t_j)$ is a discrete-time random process describing the noise in the measurements. The components of $\underline{n}(t_j)$ are assumed to be independent and $\{\underline{n}(t_j), j = 0, 1, \dots\}$ are assumed to be white zero mean Gaussian random vectors with common variance matrix. Then the goal is to estimate the state vector at different time from the available measurements vector $\{\underline{m}(t_j), j = 0, 1, \dots\}$.

In our case, the state and measurement vectors are defined as follows.

$$\underline{s}(t) = \begin{bmatrix} \frac{X(t)}{Z(t)} & \frac{Y(t)}{Z(t)} & \frac{X^{(1)}(t)}{Z(t)} & \frac{Y^{(1)}(t)}{Z(t)} & \frac{Z^{(1)}(t)}{Z(t)} & \dots \\ \frac{X^{(d_m)}(t)}{Z(t)} & \frac{Y^{(d_m)}(t)}{Z(t)} & \frac{Z^{(d_m)}(t)}{Z(t)} & \frac{Z(0)}{Z(t)} \end{bmatrix}, \quad (2.6)$$

$$\underline{m}(t_j) = \begin{bmatrix} x(t_j) \\ y(t_j) \end{bmatrix} + \underline{n}(t_j). \quad (2.7)$$

The total number of states is $(3d_m + 3)$ and the total number of measurements at each instant is 2. The evolution of the states can be found as follows. For s_1, s_2 and s_{3d_m+3} ,

$$\frac{d}{dt} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} s_3 \\ s_4 \end{bmatrix} - \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} s_5, \quad \frac{d}{dt} s_{3d_m+3} = -s_{3d_m+3} s_5. \quad (2.8)$$

For the rest of states, let $p = 3 + 3(i - 1)$ for $i = 1, \dots, d_m - 1$,

$$\frac{d}{dt} \begin{bmatrix} s_p \\ s_{p+1} \\ s_{p+2} \end{bmatrix} = \begin{bmatrix} s_{p+3} \\ s_{p+4} \\ s_{p+5} \end{bmatrix} - \begin{bmatrix} s_p \\ s_{p+1} \\ s_{p+2} \end{bmatrix} s_5, \quad (2.9a)$$

$$\frac{d}{dt} \begin{bmatrix} s_{3d_m} \\ s_{3d_m+1} \\ s_{3d_m+2} \end{bmatrix} = - \begin{bmatrix} s_{3d_m} \\ s_{3d_m+1} \\ s_{3d_m+2} \end{bmatrix} s_5. \quad (2.9b)$$

The measurement equation is rather simple. From the definitions of state and measurement vectors in (2.6) and (2.7), and using (2.2), the components of $\underline{h}(\underline{s}(t))$ at time t , h_i , are given by

$$h_1 = s_1, \quad h_2 = s_2. \quad (2.10a,b)$$

Thus, the solution for the motion of particle can be obtained by solving the above nonlinear state estimation problem. Extended Kalman filter, iterative extended Kalman filter and nonlinear filter [Mayb82] are commonly used for solving this type of problem recursively with different computational complexities. Note that the measurement equation in (2.10) is linear.

3. PERFORMANCE ANALYSIS FOR 2-D MOTION UNDER MODEL MISMATCH

Parameter jumping occurs when the actual motion switches its parameters abruptly during the observation period. Undermodeling and overmodeling occur when the order of the power series of the particle trajectory is less than and larger than, respectively, that of the actual motion. Since we do not know the exact order of the particle motion, it is worth analyzing the performance of these two problems in detail.

Since the estimation problem of the particle motion from perspective measurements is nonlinear, there is no closed-form solution in general. In order to understand the details of the performance degradation due to the model mismatch, in this section, we assume that the particle moves on a plane parallel to the image plane. For this kind of "2-D motion", we will be able to find the analytic closed-form solutions for the motion parameters and the corresponding mean-squares error. For the general 3-D motion discussed in section 2, we will argue that, based on a number of simulations conducted in section 5, the estimation have the similar behavior as that of the 2-D motion.

For the 2-D motion, the analysis can be reduced further to the following 1-D motion problem since the depth does not change and the states related to the X and Y coordinates are decoupled.

Let $m(t)$ be the positional measurement at time t of the trajectory of a moving particle in one dimension. Assume that the actual trajectory $X_s(t)$ is piecewise differentiable up to d_s -th order. The symbol 's' denotes the actual signal (trajectory). Let $X_s^{[m]}(t)$ be the m -th derivative of $X_s(t)$. Let Δ be the time separation between measurement samples. Suppose that J total samples of $m(t)$ at time $t_j = j\Delta$, $j = 0, 1, \dots, J$, are available, and that we model them as

$$m(t_j) = X_s(t_j) + n(t_j), \quad (3.1)$$

where $n(t_j)$ is a discrete-time random process describing the noise in the measurements. The $\{n(t_j)\}_{j=0}^{J-1}$ is assumed to be white zero mean Gaussian random sequence with common variance σ^2 . Then our goal is to find the estimates $\hat{X}_s^{[m]}(t)$ of $X_s^{[m]}(t)$, $m = 0, \dots, d_s$ at time t_{j-1} , optimally, in the sense of the minimum mean-squares error (MMSE). The subscript 'J' in $\hat{X}_s^{[m]}(t)$ indicates that we shall use the measurements $\{m(t_j)\}_{j=0}^{J-1}$ in the estimation process.

In general, we do not know the exact form of actual trajectory $X_s(t)$. However, if we assume that the particle moves smoothly, we may model $X_s(t)$ as a finite order power series;

$$X(t) = \sum_{p=0}^{d_m} X^{[p]}(t_0) \frac{(t-t_0)^p}{p!}, \quad (3.2)$$

where d_m denotes the order of the trajectory model $X(t)$, and $X^{[p]}(t_0)$ is the p -th derivative of $X(t)$ at time t_0 . Since it is almost impossible to derive the closed-form analytical solution for the estimate $\hat{X}_s^{[m]}(t)$ of $X_s^{[m]}(t)$ when the order of the power series is greater than three, we opt to use orthogonal polynomials to describe the actual and model trajectories.

Let $\{\xi_{p,J}(u)\}_{p=0}^{J-1}$ be a set of orthogonal polynomials with respect to the discrete points $u = 1, 2, \dots, J$. The function $\xi_{p,J}(u)$ is a p -degree polynomial and its factorial representation is given by [Rals65] as

$$\xi_{p,J}(u) = d_{0,p,J} + \sum_{k=1}^p d_{k,p,J} (u-1)(u-2)\dots(u-k), \quad (3.3)$$

where

$$d_{k,p,J} = \frac{(-1)^{p+k} (p+k)! (J-k-1)! (p!)^2}{(2p)! (p-k)! (J-p-1)! (k!)^2}. \quad (3.4)$$

Note that the leading coefficient of $\xi_{p,J}(u)$ is equal to one, i.e., $d_{p,p,J} = 1$. The orthogonal polynomials satisfy the following orthogonality property:

$$\sum_{u=1}^J \xi_{p,J}(u) \xi_{q,J}(u) = S(p,J) \delta_{p,q}, \quad (3.5)$$

where

$$S(p,J) = \frac{(p!)^4 \prod_{r=p}^{J-1} (J-r)}{(2p)! (2p+1)!}, \quad (3.6)$$

and $\delta_{p,q}$ is the Kronecker delta. Using the definitions of $\xi_{p,J}(u)$ in (3.3) and $S(p,J)$ in (3.6), one may show the following properties which will be used in the later discussion [Iu90a].

Proposition 1

(a) For large u and $m \leq p \leq J$, the m -th derivatives of $\xi_{p,J}(u)$ can be approximated as

$$\xi_{p,J}^{[m]}(u) \approx \frac{p!}{(p-m)!} u^{p-m}.$$

(b) For large J and $m \leq p \leq J$,

$$\frac{[\xi_{p,J}^{[m]}(J)]^2}{S(p,J)} \approx \frac{(2p)! (2p+1)!}{(p!)^2 ((p-m)!)^2} J^{-(2m+1)}.$$

□

Now we can express $X(t)$ in (3.2) alternatively in terms of the orthogonal polynomial as follows.

$$X(t) = \sum_{p=0}^{d_m} \theta_{p,J} \xi_{p,J}(u(t)), \quad (3.7)$$

where the argument of $\xi(\cdot)$ is the normalize-and-shift function

$$u(t) = t / \Delta + 1. \quad (3.8)$$

The m -th derivative of the $X(t)$ is given by

$$X^{[m]}(t) = \frac{1}{\Delta^m} \sum_{p=m}^{d_m} \theta_{p,J} \xi_{p,J}^{[m]}(u(t)), \quad (3.9)$$

where $\xi_{p,J}^{[m]}(u(t))$ stands for the m -th derivatives of $\xi_{p,J}(u)$ with respect to u evaluated at $u = u(t)$.

If the trajectory model $X(t)$ matches to the actual trajectory $X_s(t)$, i.e., $X(t) = X_s(t)$, then the minimum variance unbiased (MVUB) estimates $\hat{X}_s^{[m]}(t)$ of $X_s^{[m]}(t)$, and the corresponding covariances are given by [Peeb70] as

$$\hat{X}_s^{[m]}(t) = \sum_{j=0}^{J-1} w_{m,j,J}(t) m(t_j), \quad (3.10)$$

where

$$w_{m,j,J}(t) = \frac{1}{\Delta^m} \sum_{p=m}^{d_m} \frac{\xi_{p,J}^{[m]}(u(t)) \xi_{p,J}(j+1)}{S(p,J)}, \quad (3.11)$$

$$V_{m,n,J}(t) = \frac{\sigma^2}{\Delta^{m+n}} \sum_{p=m}^{d_m} \frac{\xi_{p,J}^{[m]}(u(t)) \xi_{p,J}^{[n]}(u(t))}{S(p,J)}. \quad (3.12)$$

It can be shown that the estimates $\hat{X}_s^{[m]}(t)$ in (3.10) are equal to the MMSE estimates, because of the earlier assumption of Gaussian noise in the measurement [Mayb82]. Also, these estimates can be obtained recursively by formulating the problem as a linear state estimation problem and using the Kalman filter. Since the relation of the coefficients $\{X^{[p]}(t_0)\}_{p=0}^{d_m}$ in (3.2) and $\{\theta_{p,J}\}_{p=0}^{d_m}$ in (3.7) satisfies a linear invertible transformation, one can show that the MMSE estimators of $X_s^{[m]}(t)$ for the trajectory models in terms of power series and the orthogonal polynomials are the same [Iu90a].

Note that if $X(t) = X_s(t)$, then $X^{[m]}(t) = X_s^{[m]}(t)$, and we can verify that the estimates $\hat{X}_s^{[m]}(t)$ of $X_s^{[m]}(t)$ are unbiased by using (3.1), (3.7) and (3.6);

$$\begin{aligned} E[\hat{X}_s^{[m]}(t)] &= E\left[\sum_{j=0}^{J-1} w_{mj, J}(t) m(t_j)\right] \\ &= \sum_{j=0}^{J-1} \left[\frac{1}{\Delta^m} \sum_{p=m}^{d_m} \frac{\xi_{p, J}^{[m]}(u(t)) \xi_{p, J}(j+1)}{S(p, J)} \right] \left[\sum_{q=0}^{d_m} \theta_{q, J} \xi_{q, J}(j+1) \right] \\ &= \frac{1}{\Delta^m} \sum_{p=m}^{d_m} \sum_{q=0}^{d_m} \frac{\xi_{p, J}^{[m]}(u(t)) \theta_{q, J}}{S(p, J)} S(q, J) \delta_{p, q} \\ &= \frac{1}{\Delta^m} \sum_{p=m}^{d_m} \theta_{p, J} \xi_{p, J}^{[m]}(u(t)) \\ &= X_s^{[m]}(t). \end{aligned} \quad (3.13)$$

Also, the covariance $V_{mn, J}(t)$ in (3.12) is equal to the error correlation

$$R_{mn, J}(t) = E\left[\left[\hat{X}_s^{[m]}(t) - X_s^{[m]}(t)\right] \left[\hat{X}_s^{[n]}(t) - X_s^{[n]}(t)\right]\right]. \quad (3.14)$$

Note that there is difference between $V_{mn, J}(t)$ and $R_{mn, J}(t)$. The former is the covariance of the estimate $\hat{X}_s^{[m]}(t)$ whereas the latter is the correlation between the errors $[\hat{X}_s^{[m]}(t) - X_s^{[m]}(t)]$ and $[\hat{X}_s^{[n]}(t) - X_s^{[n]}(t)]$. The auto-correlation $R_{mm, J}(t)$ is the mean-squares error of the estimate $\hat{X}_s^{[m]}(t)$.

Whenever the trajectory model does not describe the actual trajectory, the above properties are not valid in general, i.e., the estimator $\hat{X}_s^{[m]}(t)$ may have the bias

$$B_{m, J}(t) = E[\hat{X}_s^{[m]}(t)] - X_s^{[m]}(t), \quad (3.15)$$

and the error correlation becomes

$$R_{mn, J}(t) = V_{mn, J}(t) + B_{m, J}(t) B_{n, J}(t). \quad (3.16)$$

The bias $B_{m, J}(t)$ and the mean-squares error $R_{mm, J}(t)$ are two useful quantities for monitoring the performance of the estimators. The former indicates how close the estimate $\hat{X}_s^{[m]}(t)$ in average is to the true value $X_s^{[m]}(t)$, while the latter measures how large the error in the estimate is, in average.

3.1 Model Mismatch due to Parameter Jumping

Suppose a particle which undergoes a smooth motion changes its motion abruptly at time t_{sw} such that its true trajectory is given as

$$X_s(t) = \begin{cases} X_{s1}(t) = \sum_{p=0}^{d_1} \theta_{s1p, J} \xi_{p, J}(u(t)) & 0 \leq t < t_{sw} \\ X_{s2}(t) = \sum_{p=0}^{d_2} \theta_{s2p, J} \xi_{p, J}(u(t)) & t_{sw} \leq t < t_{J-1} \end{cases}, \quad (3.17)$$

where d_s is the order of the trajectory. If we use the d_m -degree polynomial $X(t)$ in (3.7) to model $X_s(t)$ and assume $d_m = d_s$, then, from (3.10), the mean of the estimate $\hat{X}_s^{[m]}(t)$ of $X_s^{[m]}(t)$ are given by

$$\begin{aligned} E[\hat{X}_s^{[m]}(t)] &= E\left[\sum_{j=0}^{J-1} w_{mj, J}(t) m(t_j)\right] \end{aligned}$$

$$\begin{aligned} &= \sum_{j=0}^{[sw]-1} w_{mj, J}(t) X_{s1}(t_j) + \sum_{j=[sw]}^{J-1} w_{mj, J}(t) X_{s2}(t_j) \\ &= \sum_{j=0}^{J-1} w_{mj, J}(t) X_{s2}(t_j) + \sum_{j=0}^{[sw]-1} w_{mj, J}(t) [X_{s1}(t_j) - X_{s2}(t_j)], \end{aligned} \quad (3.18)$$

where $[sw]$ is the smallest integer which is greater than or equal to sw . Similarly to the derivation of (3.13), one may show that the first term in the right hand side of (3.18) is equal to $X_s^{[m]}(t)$ if $t \geq t_{sw}$. So the bias of the estimate at time t_{j-1} is:

$$B_{m, J}(t_{j-1}) = \sum_{j=0}^{[sw]-1} w_{mj, J}(t_{j-1}) [X_{s1}(t_j) - X_{s2}(t_j)], \quad (3.19)$$

and the error correlation of the estimate $R_{mn, J}(t_{j-1})$ is evaluated from (3.16) with the above $B_{m, J}(t_{j-1})$. Note that the bias is independent of the noise.

The bias $B_{m, J}(t_{j-1})$ and the mean-squares error $R_{mm, J}(t_{j-1})$ decrease as more frames are involved in the estimation process. It may take a large number of frames until the bias and mean squares error are reduced to a manageable size, but we are likely to lose track of the particle if another parameter jump occurs before the process settles down. To illustrate this situation, let us consider a simple example. Suppose we use a constant value model to estimate a piecewise constant trajectory, i.e., $d_m = d_s = 0$, for which the particle switches its position from $\theta_{s10, J}$ to $\theta_{s20, J}$ abruptly at time $(sw \Delta)$, where sw is an integer. Then the bias and the mean-squares error of the estimate $\hat{X}_s^{[0]}(t_{j-1})$ can be found as

$$B_{0, J}(t_{j-1}) = \frac{sw}{J} (\theta_{s10, J} - \theta_{s20, J}), \quad (3.20)$$

$$R_{00, J}(t_{j-1}) = \frac{\sigma^2}{J} + \left[\frac{sw}{J} \right]^2 (\theta_{s10, J} - \theta_{s20, J})^2. \quad (3.21)$$

The first term in the right hand side of (3.21) is the error due to the noise, and the second term is the extra error due to the parameter jumping. Although this bias $B_{0, J}(t_{j-1})$ and the mean-squares error $R_{00, J}(t_{j-1})$ will decrease as J increases, a large number of J is required to suppress it if sw is large. For example, if we want the bias $B_{0, J}(t_{j-1})$ to be $(\theta_{s10, J} - \theta_{s20, J})/2$, then we must wait until J becomes equal to $(2 sw)$. It means that, after the particle switches its value from $\theta_{s10, J}$ to $\theta_{s20, J}$ at time $(sw \Delta)$, we must wait another total sw samples to get an estimate whose value, in average, equals to the average of $\theta_{s10, J}$ and $\theta_{s20, J}$. This is because the estimation process has memorized all the past 'invalid' measurements. Therefore a large number of new measurements are required in order to 'nullify' these invalid values.

3.2 Model Mismatch due to Undermodeling

Suppose that the actual trajectory $X_s(t)$ is described by

$$X_s(t) = \sum_{p=0}^{d_s} \theta_{sp, J} \xi_{p, J}(u(t)), \quad (3.22)$$

then $X_s^{[m]}(t)$ which we would like to estimate is given as

$$X_s^{[m]}(t) = \frac{1}{\Delta^m} \sum_{p=m}^{d_s} \theta_{sp, J} \xi_{p, J}^{[m]}(u(t)). \quad (3.23)$$

For the trajectory model in (3.7), the estimate $\hat{X}_s^{[m]}(t)$ of $X_s^{[m]}(t)$ and the covariance $V_{mn, J}(t)$ are given in (3.10) and (3.12), respectively. If the model is perfect, i.e., $d_m = d_s$, then these estimates are

optimal. The estimation problems are called undermodeling and overmodeling if $d_m < d_s$ and $d_m > d_s$, respectively. We will analyze the performance of the estimators of (3.10) for the undermodeling first.

For the actual trajectory in (3.22) and the model of trajectory in (3.7), we can find the mean of estimates $\hat{X}_f^{[m]}(t)$ of $X_f^{[m]}(t)$, similarly to the derivation in (3.13), as follows:

$$E[\hat{X}_f^{[m]}(t)] = \frac{1}{\Delta^m} \sum_{p=m}^{d_m} \theta_{sp,J} \xi_{sp}^{[m]}(u(t)). \quad (3.24)$$

From (3.15), (3.23), (3.16) and (3.12), the bias and the error correlation at t_{j-1} are obtained as

$$B_{m,J}(t_{j-1}) = -\frac{1}{\Delta^m} \sum_{p=d_m+1}^{d_s} \theta_{sp,J} \xi_{sp}^{[m]}(J), \quad (3.25)$$

$$R_{m,n,J}(t_{j-1}) = \frac{\sigma^2}{\Delta^{m+n}} \sum_{p=m}^{d_m} \frac{\xi_{sp}^{[m]}(J) \xi_{sp}^{[n]}(J)}{S(p,J)} + B_{m,J}(t_{j-1}) B_{n,J}(t_{j-1}). \quad (3.26)$$

The mean-squares error of the estimates $\hat{X}_f^{[m]}(t)$ at time t_{j-1} becomes

$$R_{mm,J}(t_{j-1}) = \frac{\sigma^2}{\Delta^{2m}} \sum_{p=m}^{d_m} \frac{[\xi_{sp}^{[m]}(J)]^2}{S(p,J)} + \frac{1}{\Delta^{2m}} \left[\sum_{p=d_m+1}^{d_s} \theta_{sp,J} \xi_{sp}^{[m]}(J) \right]^2. \quad (3.27)$$

The first term in the right hand side of (3.27) is proportional to the common variance of the noise. From Proposition 1(b), each term in the summation will converge to zero at the rate of $J^{-(2m+1)}$ as J increases. The second term in (3.27) reveals the additional error due to the undermodeling. From Proposition 1(a), $\xi_{sp}^{[m]}(J) \approx \frac{p!}{(p-m)!} J^{p-m}$. Consequently, as J increases, both the additional error term and the magnitude of the bias in (3.25) will, in general, increase without a bound!

3.3 Model Mismatch due to Overmodeling

In the following, we will show that, in the case of overmodeling, the estimate is unbiased but the mean-squares error increases as the order of the model d_m increases.

Similarly to the discussion on the undermodeling, we can show that the mean of estimates $\hat{X}_f^{[m]}(t)$ for the actual trajectory (3.22) is given by

$$E[\hat{X}_f^{[m]}(t)] = \frac{1}{\Delta^m} \sum_{p=m}^{d_m} \sum_{q=0}^{d_s} \frac{\xi_{sp}^{[m]}(u(t)) \theta_{sq,J}}{S(p,J)} S(q,J) \delta_{p,q}. \quad (3.28)$$

Since $d_m > d_s$ and using (3.2.2),

$$\begin{aligned} E[\hat{X}_f^{[m]}(t)] &= \frac{1}{\Delta^m} \sum_{p=m}^{d_m} \theta_{sp,J} \xi_{sp}^{[m]}(u(t)) \\ &= X_f^{[m]}(t). \end{aligned} \quad (3.29)$$

Thus, the estimates are unbiased and the error correlation $R_{m,n,J}(t)$ is equal to the covariance of estimates $V_{m,n,J}(t)$, i.e.,

$$R_{m,n,J}(t) = \frac{\sigma^2}{\Delta^{m+n}} \sum_{p=m}^{d_m} \frac{\xi_{sp}^{[m]}(u(t)) \xi_{sp}^{[n]}(u(t))}{S(p,J)}. \quad (3.30)$$

This implies that the mean-squares error $R_{mm,J}(t)$ of $\hat{X}_f^{[m]}(t)$ at time t_{j-1} is given by

$$\begin{aligned} R_{mm,J}(t_{j-1}) &= \frac{\sigma^2}{\Delta^{2m}} \sum_{p=m}^{d_m} \frac{[\xi_{sp}^{[m]}(J)]^2}{S(p,J)} \\ &= \frac{\sigma^2}{\Delta^{2m}} \sum_{p=m}^{d_s} \frac{[\xi_{sp}^{[m]}(J)]^2}{S(p,J)} + \frac{\sigma^2}{\Delta^{2m}} \sum_{p=d_s+1}^{d_m} \frac{[\xi_{sp}^{[m]}(J)]^2}{S(p,J)}. \end{aligned} \quad (3.31)$$

The first term in the right hand side of (3.31) is equal to the minimum mean-squares error if the model matches to the true trajectory. By using Proposition 1(b), each term in the summation converges to zero at the rate of $J^{-(2m+1)}$ as J increases. The second term is the additional error due to the overmodeling. Since $S(p,J)$ is greater than zero for $J > d_m > d_s$, this additional error is positive and will increase as d_m increases. Fortunately, this error converges to zero at the rate of $J^{-(2m+1)}$ as J increases.

4. FLAT MMF

In this section, we propose the new filter FLAT MMF to solve the problem of the model mismatches we have discussed so far. Section 4.1 discusses the motivation of this filter. Section 4.2 reviews the multiple model filter and describes the basic structure of FLAT MMF. Section 4.3 summarizes some properties of FLAT MMF regarding the model mismatches.

4.1 Motivation of FLAT MMF

Let us consider the problem of parameter jumping first. As we discussed in section 3.1, the reason that the estimates of the parameter jumping have a very large biases and mean-squares errors after the particle switches its value is that the filter has memorized many 'invalid' measurements. Thus, we would like to design a filter in which part or all of these 'invalid' measurements are suppressed. For the example presented in section 3.1, one may have observed that if we had started another filter some time after the first one, then the estimate from this filter would have had smaller bias and mean-squares error. In the extreme case, if the filter is started after the switch time, then the estimate will be unbiased and have the minimum mean-squares error. To enlighten this discussion further, let us consider a simple experiment.

Experiment 1

Consider a particle moving from (5, 5, 20) units to (-4, 2, 20) units with velocity (-4.16, -1.38, 0) units/second then moving to (0, -5, 20) units with velocity (2.176, -3.81, 0) units/second. Thus, there is no depth change in the motion, and the velocity is piecewise constant. Assume that we use a constant velocity model and that we start the identical estimators at times 0, 20, 40, 60 and 80 samples. Figure 1a depicts the estimates of velocity $X^{(1)}(t)/Z(0)$. (Detail of the experiment setup and the procedure for finding the estimate will be discussed in section 5). As we expected, the model error due to the velocity jump makes the estimates possess a very large error. Filters that started after the velocity jump yield the better estimate. On the other hand, the older filter has the better noise suppression than the younger one. □

The above observation may suggest that we restart the filter every few samples in order to keep the number of 'invalid' measurements small. But this suggestion does not work because the filter needs a large number of samples to find the estimates and to suppress the noise. However, if we use two or more filters which

are started (triggered) at different instants and combine the estimates from these filters adequately, then we may be able to obtain the estimates which have a good noisy suppression and avoid the problem of parameter jumping. It is because that the older filter provides the estimates from more measurements while the younger one memorize less 'invalid' measurements. Moreover, we may restart the oldest filter again once it becomes too old, say after 100 samples, because the extra old measurements memorized in this filter become out of date. The above discussion motivates us to propose the FLAT MMF. We will discuss it in next section in more detail.

The proposed FLAT MMF may also solve the problem of undermodeling. As we discussed in section 3.2, if we use a low-order model to estimate a high-order trajectory, the estimates will possess biases and these biases increase without a bound in general. However, we observe that these biases are small around the time the filter is started because the actual trajectory can be approximated adequately by the model within a small neighborhood. The following experiment is conducted to illustrate this observation.

Experiment 2

Suppose a particle moves with a constant acceleration, but without a depth change. Further suppose that we use the constant velocity model. The initial position of the particle is $(-6.8, -6.8, 20)$ units. It moves with velocity $[3.4 \ 10 \ 0]^T + [0 \ -5 \ 0]^T t$ units/second. Figure 2a shows the estimates of $Y^{(1)}(t) / Z(0)$ as the identical filters are started at different times. From this figure, it is obvious that the model error due to undermodeling makes the estimates diverge. However, each filter gives reasonably good estimates during a short interval right after it is started. \square

For the overmodeling, the opposite is true. Recall that the estimates for overmodeling are unbiased and the extra mean-squares errors come from the trajectory model that provides 'over-freedom'. If the trajectory model is fixed, then the only way to achieve the better estimate, i.e., less mean-squares errors is to use more measurements. Consequently, the estimates from the filter started later will have larger mean-squares errors because it has used less measurements. However, if one could conceive a mechanism that combines the estimates from multiple filters in such a way that the final estimates are dominated by those from the oldest filter, then the estimate obtained from the new filter does not degrade significantly, compared to the estimates obtained from a single filter. Note that, due to the numerical stability and the computational cost, the order of the trajectory model can not be high. This is especially true for estimating 3-D object motion from perspective measurements because the size of the state vector equals at least three times of the order of the trajectory model being used [Broi86b]. Thus, the problem of overmodeling is less serious than the problem of undermodeling and parameter jumping.

Up to this point, we observe that we may solve the problem of parameter jumping, undermodeling and overmodeling all at once, by using a number of filters triggered at different instants, provided that we combine the estimates from these filters adequately. Thus, one may raise two questions: How do we combine these estimates adequately and what do we mean by 'adequately'? The multiple model filter discussed in next section provides an answer to these questions.

4.2 Basic Structure of FLAT MMF

In this section, we will first review briefly the multiple model filter (MMF) and then describe the basic structure of the proposed FLAT MMF. The idea of multiple model filter is first proposed by Magill in 1965 for estimating the state of system with uncertainty [Magi65]. Since then, numerous applications and researches on the behavior of MMF have been conducted. The detail of MMF can be obtained from the references in [Mayb82]. Only a brief description of MMF is given here.

Suppose we want to estimate the state $\underline{s}(t)$ at time t_i of a system of interest from the measurements $\{\underline{m}(t_j)\}_{j=0}^{i-1}$. Assume that this state estimation problem can be modeled adequately by the one in which the plant equation and measurement equation are linear, except there are some uncertainties in the modeling, such as the covariance matrices of the model noise and measurement noise, and some parameters defining the state transition matrix. Let \underline{a} denote the vector of these uncertain parameters and assume that \underline{a} belongs to the set of values $\{\underline{a}_k\}_{k=1}^K$. Then, for each \underline{a}_k , we may construct a Kalman filter, based upon the model associated with \underline{a}_k , to estimate the state. These K filters are independent to each other, and thus they can be processed simultaneously. The final estimate of the state is obtained by combining the estimates from these K filters. The state estimation based on the above structure is called multiple model filtering. It can be shown that for the above MMF structure, the optimal state estimate, in the sense of MMSE, is given by [Mayb82]

$$\hat{\underline{s}}(t_i^+) = \sum_{k=1}^K \hat{\underline{s}}_k(t_i^+) p_k(t_i) \quad (4.1)$$

where $\hat{\underline{s}}_k(t_i^+)$ is the state estimate produced by k -th Kalman filter based on the assumption that the parameter vector equals \underline{a}_k , $p_k(t_i)$ is the hypothesis conditional probability and

$$p_k(t_i) = \text{Prob}\{\underline{a} = \underline{a}_k \mid \underline{M}_{0,i-1}\} \\ = \frac{f(\underline{m}(t_i) \mid \underline{a}_k, \underline{M}_{0,i-1}) p_k(t_{i-1})}{\sum_{j=1}^K f(\underline{m}(t_i) \mid \underline{a}_j, \underline{M}_{0,i-1}) p_j(t_{i-1})}, \quad (4.2)$$

where $\underline{M}_{0,i-1}$ is the composite vector which comprises the measurement $\underline{m}(t)$ from t_0 to t_{i-1} , i.e.,

$$\underline{M}_{0,i-1} = [\underline{m}^T(t_0) \ \underline{m}^T(t_{i-1}) \ \cdots \ \underline{m}^T(t_1)]^T.$$

The covariance of $\hat{\underline{s}}(t_i^+)$ is

$$V(t_i^+) = \sum_{k=1}^K p_k(t_i) \{ V_k(t_i^+) + [\hat{\underline{s}}_k(t_i^+) - \hat{\underline{s}}(t_i^+)] [\hat{\underline{s}}_k(t_i^+) - \hat{\underline{s}}(t_i^+)]^T \} \quad (4.3)$$

where $V_k(t_i^+)$ is the covariance of $\hat{\underline{s}}_k(t_i^+)$ computed by the k -th Kalman filter. The conditional probability $f(\underline{m}(t_i) \mid \underline{a}_k, \underline{M}_{0,i-1})$ in (4.2) is evaluated as

$$f(\underline{m}(t_i) \mid \underline{a}_k, \underline{M}_{0,i-1}) = \frac{\exp\{-\frac{1}{2} \underline{r}_k^T(t_i) A_k^{-1}(t_i) \underline{r}_k(t_i)\}}{(2\pi)^{r_m/2} |A_k(t_i)|^{1/2}}, \quad (4.4)$$

where

$$A_k(t_i) = H V_k(t_i^-) H^T + R, \quad (4.5)$$

$$\underline{r}_k(t_i) = \underline{m}(t_i) - H \hat{\underline{s}}_k(t_i^-), \quad (4.6)$$

and $\hat{x}_k(t_i^-)$, $V_k(t_i^-)$ and $\hat{r}_k(t_i)$ are the state estimate, the covariance and the residual at t_i of the k -th Kalman filter, based upon the measurements $\{\underline{m}(t_j)\}_{j=0}^{i-1}$, respectively. R , H and n_m are the covariance matrix of measurement noise, the measurement matrix and the total number of measurements at each time, respectively. Note that $A_k(t_i)$ and $\hat{r}_k(t_i)$ are available as intermediate results of the k -th Kalman filter. Thus, the conditional probability $f(\underline{m}(t_i) | \underline{a}_k, M_{0,i-1})$ as well as the weighting factors $p_k(t_i)$ can be obtained by a small amount of extra computation.

In summary, the MMF is composed of a bank of K separate Kalman filters, each based on a particular parameter vector \underline{a}_k . The overall state estimate is the linear combination of the state estimates generated by these Kalman filters. The weighting factors $p_k(t_i)$ is updated recursively as (4.2), using the current $A_k(t_i)$ and $\hat{r}_k(t_i)$. The block diagram of the MMF is depicted in figure 3. All the filters are run simultaneously and the extra computation for updating the weighting factors $p_k(t_i)$ compared to the normal Kalman filter is negligible.

The basic structure of the FLAT MMF we are proposing is composed of a set of K identical Kalman filters, each triggered at different time. The overall state estimate is the probabilistically weighted average of the state estimates generated by these Kalman filters, as for the MMF. Without loss of generality, we assume that the k -th filter is triggered at time $(k-1)J\Delta$, where J is an integer. Each filter will die after $(K-J)\Delta$ seconds and then it will be triggered again, i.e., each filter only has a 'lifetime' of $(K-J)\Delta$. Figure 4 shows the timing of the FLAT MMF for $K=4$. At each instant, in general, there are K filters being processed simultaneously. Thus, a FLAT MMF is a MMF in which all the filters are identical but have the different starting time, i.e., the uncertain parameter vector \underline{a} discussed before is the time that the filter is started. For the nonlinear state estimation problem such as the problem of motion estimation discussed in section 2, FLAT MMF can still be used to estimate the state if the Kalman filter is replaced by the extended Kalman filter or some other nonlinear filter. Furthermore, we may include other uncertain parameters into the estimation process by replacing each filter with a MMF concerning on those uncertain parameters.

The key feature of the proposed FLAT MMF is that the multiple, asynchronous filters operate on the different sets of past measurements. Hence, as we discussed in section 4.1, the estimates from these filters contain the one that has good noisy suppression, the one that contains small number or none of 'invalid' measurements, and the one that has small model error. The structure of MMF provides a way to combine these estimates appropriately, in the sense of MMSE, so that the 'correct' estimate will appear at the final estimate. The behavior of FLAT MMF for model mismatch is discussed further in next sub-section. Another feature of FLAT MMF is that all the filters can be processed simultaneously and the computational cost for combining the estimates from the filters is relatively small. Thus, the FLAT MMF can be implemented efficiently for real-time applications.

4.3 FLAT MMF and Model Mismatch

For the 2-D motion discussed in section 3.1, we have analyzed in detail the behavior of the estimation for these three model mismatches [Iu90a]. Due to the space limitation, we only summar-

ize the result as follows.

- (1) For the parameter jumping, the overall estimates before the feature point changes its motion are dominated by the estimates from the oldest filter, which is the one that provides the most noise suppression. After the jump occurs, the overall estimates approach exponentially to the estimates produced by the youngest filter. Then after another filter is triggered, the estimates from this newly started filter get control of the overall estimates, and the estimation of FLAT MMF approaches to the one with unbiased and minimum mean-squares error estimates.
- (2) For the undermodeling, the overall estimates approach exponentially to the estimates from the newly started filter whenever a new trigger occurs. Although the overall estimates have biases, these biases are considerably smaller than those of one filter. These biases depend on the time interval between the triggers $(J\Delta)$ and the discrepancy between the trajectory model and the true trajectory.
- (3) For the overmodeling, the overall estimates are dominated by the estimates of the oldest filter. The estimation does not degrade significantly even if we use the FLAT MMF instead of the normal filter.

5. SIMULATION RESULTS

In order to demonstrate the performance degradation of the model mismatches and the performance improvement as the FLAT MMF is used, a number of experiments on simulated data are conducted. The noisy measurements of the projected position $\underline{p}(t)$, at sampling instants t_j , $j=0, 1, \dots$, are generated by adding white zero mean Gaussian noise to the exact values of $\underline{p}(t)$, which is obtained by using (2.3) and (2.2). The standard deviations of the noise are set to 2.5 pixels. The focal length of the camera is set to one unit. The visible portion of the image plane is $(-0.36, 0.36) \times (-0.36, 0.36)$ units. This corresponds to the viewing angle of ± 20 degrees. The observed image is considered as 256×256 pixels. The time interval between frames is 0.04 second. For the 3-D motion, extended Kalman filter is used to solve the nonlinear state estimation problem. For the 2-D motion discussed in section 3, the problem becomes a linear one, and the Kalman filter is used. In both cases, the initial estimates of projective position are set to their measurements and the states relating to their derivatives are set to zero. For the 3-D motion, the relative depth $Z(0)/Z(t)$ is set to one.

5.1 Simulation on 2-D Motion

Experiment 1 and 2 (continue)

For the experiments 1 and 2 discussed in section 4.1, we use the FLAT MMF with two Kalman filters. Each filter is triggered at every 25 samples. Figures 1b and 2b depict the estimates of $X^{(1)}(t)/Z(0)$ and $Y^{(1)}(t)/Z(0)$ for these experiments, respectively. These results show that the FLAT MMF works quite well in handling the model mismatches. Comparing figures 1b and 2b to figures 1a and 1b, one may find that the overall estimates of the FLAT MMF are formed as just 'cutting out' the correct portions of the estimates from two filters.

Experiment 3: parameter jumping

In this experiment, the particle moves on the plane $Z = 20$ units from the initial position $(-5, 0, 20)$ units. After six turns, the particle moves back to its starting position. The angles of these six turns are 90° , 120° , -60° , 180° , -120° and 150° . Figure 5a shows the ideal trajectory and its noisy measurement. Figures 5b compares the estimates of velocity $X^{(1)}(t) / Z(0)$ for the normal Kalman filter and for the FLAT MMF. Figure 5c compares the position estimates. The improvement is substantial both in the position and in the velocity. From these results, we observe that the estimation using one filter almost lost the track of the motion immediately after the first turn. On contrast, the FLAT MMF provides very good estimates even at the place having very sharp velocity change (180° turn).

When we re-run the FLAT MMF on the same data with increasing the velocity of particle, its performance gets worse, but the estimates are still fairly good. Figure 6a shows the estimate of $X^{(1)}(t) / Z(0)$ when the particle moves three times faster than the original speed. In order to improve the estimation, one may use more filters in the FLAT MMF. Figure 6b shows the estimate of $X^{(1)}(t) / Z(0)$ with the FLAT MMF using five filters, each filter is triggered at every ten samples.

Experiment 4: undermodeling

In this experiment, the particle moves on the plane $Z = 20$ units, along an ellipse with angular velocity 0.15 radian/seconds. The length of the principal axes of the ellipse are 12 and 8 units. The particle is at $(6, 0, 20)$ units initially. Figure 7a shows the exact and noisy trajectories of the first 160 samples. Each filter assumes that the particle moves with constant acceleration. Thus, this situation is highly undermodeled because the circular motion requires a very high-order power series to describe it properly. We have used the normal Kalman filter and the FLAT MMF with two filters, triggered at every 25 samples. Figures 7b and 7c show the estimates of velocity $X^{(1)}(t) / Z(0)$ and the prediction of trajectory, respectively. From these results, we observe that the estimation using one filter has a very large error while the estimation using FLAT MMF is quite good. Figure 8a and 8b depict the estimates of $X^{(1)}(t) / Z(0)$ using the FLAT MMF with two and five filters respectively, when the particle moves four times faster. These results show that even for such high-speed circular motion, FLAT MMF can still provide fairly good estimates.

5.2 Simulation on 3-D Motion

In the following two experiments, the motion is modeled with constant acceleration. In building the FLAT MMF, three extended Kalman filter, triggered at every 33 samples are used.

Experiment 5: parameter jumping

This experiment simulates the motion of a bouncing ball. The initial position of the particle is $(-6.5, 1, 20)$ units. It moves with velocity $[1.5 \ 5 \ 2]^T + [0 \ -2.5 \ 0]^T t$ units/second. When it hits the point $(-0.56, 1.198, 27.92)$ units, the velocity changes to $[1.5 \ 4.5 \ 2]^T + [0 \ -2.5 \ 0]^T t$. Figure 9a depicts the exact and noisy trajectories. Figures 9b and 9c show the estimates of velocity $Y^{(1)}(t) / Z(t)$ and $Z^{(1)}(t) / Z(t)$, respectively. From these results, we observe that the estimates from FLAT MMF is quite good while the estimates from using one filter is almost useless.

Experiment 6: undermodeling

In this experiment, a helix motion is simulated, the particle moves circularly while it travels in the Z -direction with constant velocity. The angular velocity is 0.07 radians/second and the Z -velocity is 1.5 units/second. The initial position of the particle is $(3.3, 3.3, 10)$ units. Figure 10a depicts the exact and noisy trajectories. Figures 10b and 10c show the estimates of velocity $X^{(1)}(t) / Z(t)$ and $Z^{(1)}(t) / Z(t)$, respectively. These results show the same conclusion the estimation using the normal filter suffers seriously by the model mismatch while the estimation using the FLAT MMF can provide much better estimates.

6. SUMMARY

In this paper, we have shown that the performance of conventional estimator would degrade significantly if the motion model did not match to the actual motion. In order to solve the model mismatch problem, we proposed the FLAT MMF, a parallel, recursive filter. Since each filter inside the FLAT MMF operates on a different set of past measurements, there must be a filter that provides the estimates with the best noise suppression, another one that estimates from the smallest number (or none) of invalid measurements, and the one that possesses the smallest model error. The FLAT MMF uses the structure of multiple model filter to combine these estimates optimally, in the sense of MMSE. Hence, the FLAT MMF is quite effective in suppressing the various adversary effects due to the modeling error. Our claim is verified by the formal analysis of the FLAT MMF on the 2-D motion of a single feature point. A numerous simulations performed on the 3-D motion of a feature point further support the effectiveness of FLAT MMF.

Reference

- [Boll85] R.C. Bolles and H.H. Bakers, "Epipolar-plane image analysis: a technique for analyzing motion sequences", in Workshop on computer vision: representation and control, Oct. 1985.
- [Broi86a] T.J. Brodia and R.Chellappa, "Estimation of object motion parameters from noisy images", IEEE PAMI, Jan 1986, pp. 90-99.
- [Broi86b] T.J. Brodia and R. Chellappa, "Kinematics and structure of a rigid object from a sequence of noisy images", IEEE workshop on motion 1986, pp. 95-100.
- [Iu89] S.-L. Iu and K. Wahn, "Estimation of 3-D motion and structure based on a temporally-oriented approach with the method of regression", IEEE Workshop on visual motion, March 1989, pp. 273-281.
- [Iu90a] S.-L. Iu, "Analysis of the Effects of Model Mismatch and FLAT MMF for Estimating Particle Motion", Grasp Lab Tech Report MS-CIS-90-10, Univ. of Pennsylvania, Feb. 1990.
- [Iu90b] S.-L. Iu, "Estimation of 3-D motion and structure from images by using temporal-based approach", Ph.D. Thesis, Elec. Engineering Department, Univ. of Pennsylvania, 1990.
- [Kuma89] R.V.R. Kumar, A. Tirumalai and R. C. Jain, "A non-linear optimization algorithm for the estimation of structure and motion parameters", IEEE Workshop on visual motion, March 1989, pp. 136-143.

[Mayb82]

P. S. Maybeck, *Stochastic models, estimation and control*, vol. 1-2, Academic Press, 1982.

[Magi65]

D. T. Magill, "Optimal adaptive estimation of sampled stochastic process", *IEEE Trans. Auto. Control*, Oct. 1965, pp. 434-439.

[Peeb70]

P. Z. Peebles, "Alternative approach to the prediction of polynomial signals in noise from discrete data", *IEEE T-AES*, July 1970, pp. 534-543.

[Rals65]

A. Ralston, *A first course in numerical analysis*, McGraw-Hill Co., 1965.

[Weng87]

J. Weng, T.S. Huang and N. Ahuja, "3-D motion estimation, understanding and prediction from noisy image sequences", *IEEE PAMI*, May 1987, pp. 370-389.

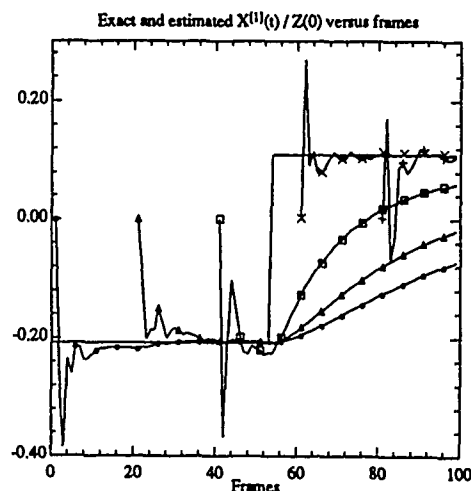


Fig. 1a. Exact and estimated $X^{(1)}(t) / Z(0)$ versus number of frames for experiment 1. Estimators are triggered at samples 0 (O), 20 (Δ), 40 (\square), 60 (\times) and 80 (+).

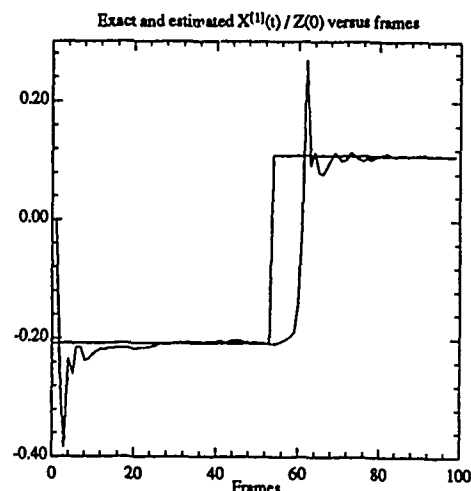


Fig. 1b. Exact and estimated $X^{(1)}(t) / Z(0)$ versus number of frames by using FLAT MMF, for experiment 1.

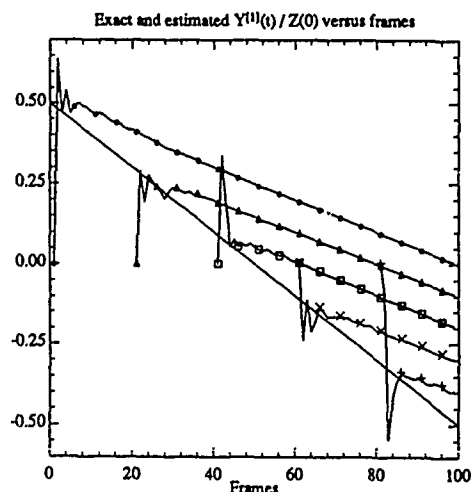


Fig. 2a. Exact and estimated $Y^{(1)}(t) / Z(0)$ versus number of frames for experiment 2. Estimators are triggered at samples 0 (O), 20 (Δ), 40 (\square), 60 (\times) and 80 (+).

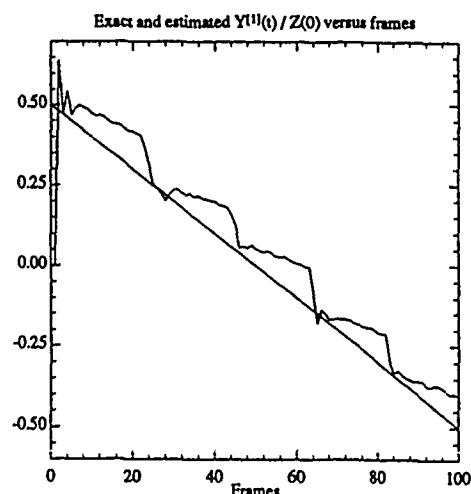


Fig. 2b. Exact and estimated $Y^{(1)}(t) / Z(0)$ versus number of frames by using FLAT MMF, for experiment 2.

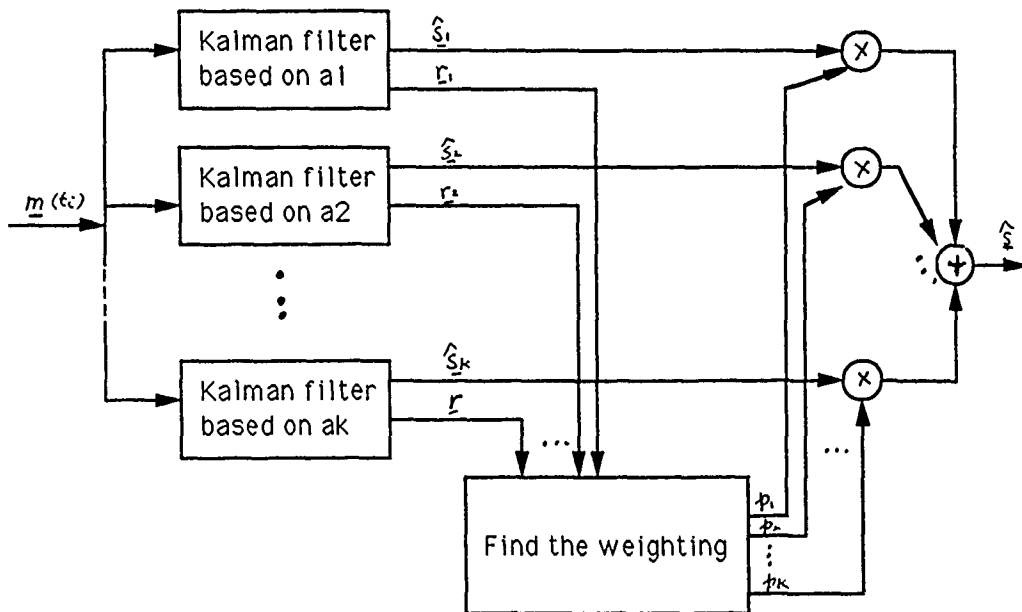


Fig. 3. Block diagram of multiple model filter

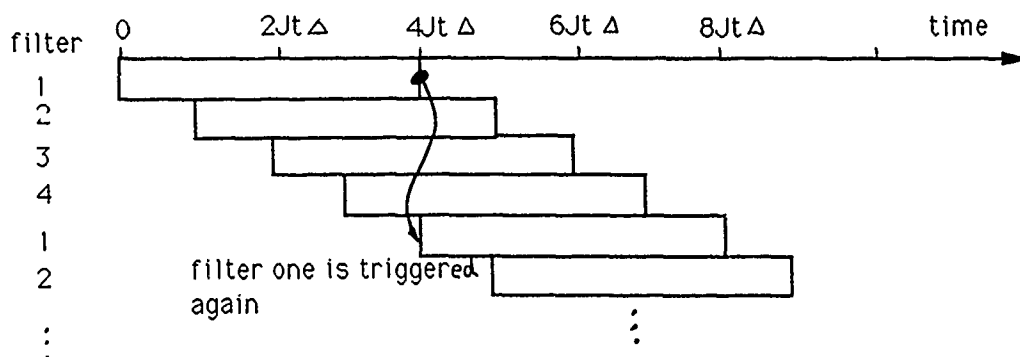


Fig. 4. Timing of FLAT MMF for K=4

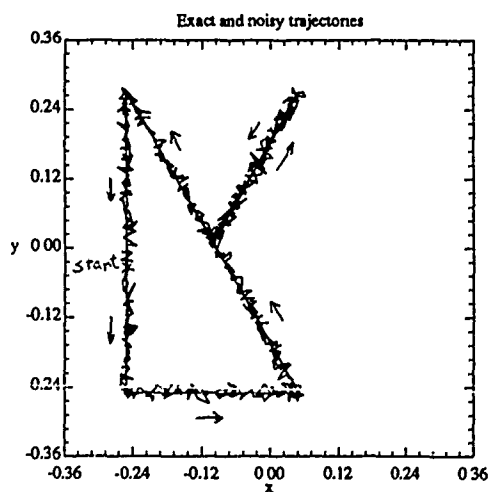


Fig. 5a. Exact and noisy trajectories for experiment 3. Standard deviation of the noise is 2.5 pixels.

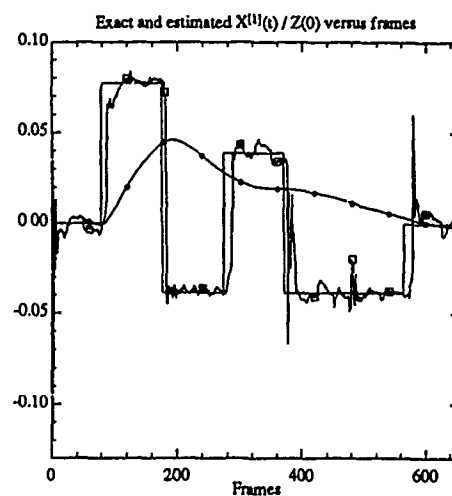
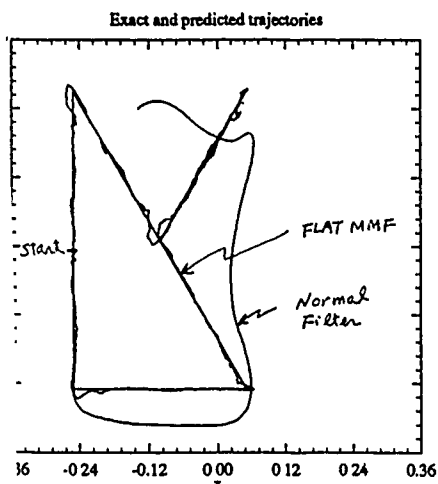
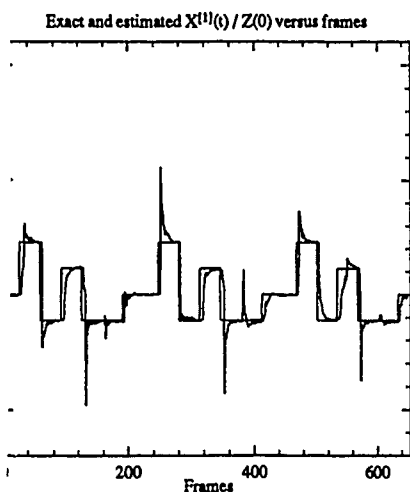


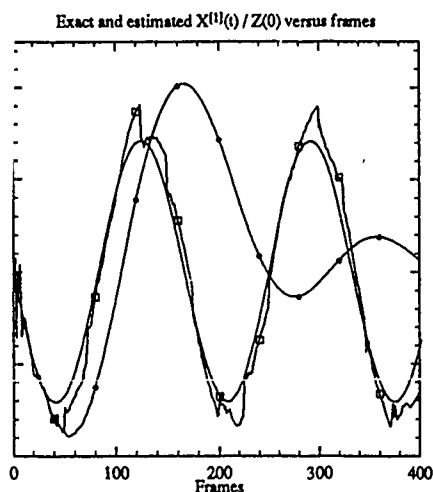
Fig. 5b. Exact and estimated $X^{(1)}(t)/Z(0)$ versus number of frames by using normal filter (O) and FLAT MMF (□), for experiment 3.



5c. Exact and predicted trajectories by using normal filter and FLAT MMF, for experiment 3.



6b. Exact and estimated $X^{(1)}(t) / Z(0)$ versus number of frames by using FLAT MMF with five filters, for experiment 3 (faster motion).



7b. Exact and estimated $X^{(1)}(t) / Z(0)$ versus number of frames by using normal filter (O) and FLAT MMF (□), for experiment 4.

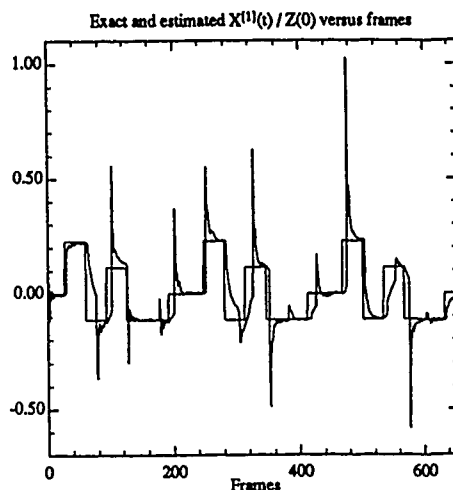


Fig. 6a. Exact and estimated $X^{(1)}(t) / Z(0)$ versus number of frames by using FLAT MMF with two filters, for experiment 3 (faster motion).

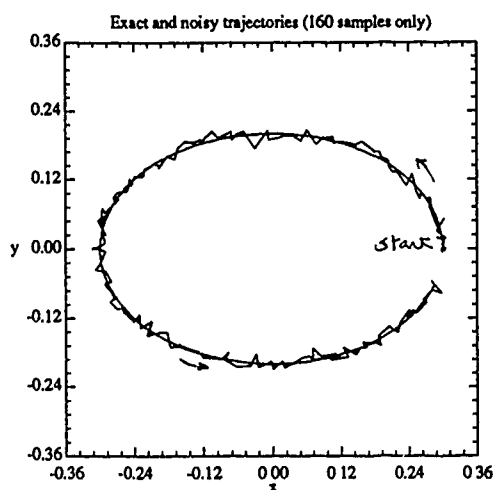


Fig. 7a. Exact and noisy trajectories for experiment 4. Standard deviation of the noise is 2.5 pixels.

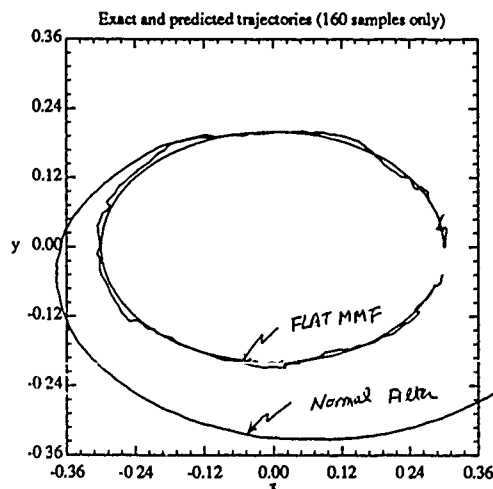


Fig. 7c. Exact and predicted trajectories by using normal filter and FLAT MMF, for experiment 4.

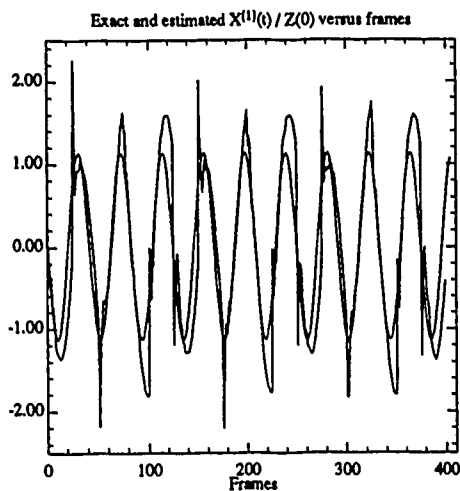


Fig. 8a. Exact and estimated $X^{(1)}(t) / Z(0)$ versus number of frames by using FLAT MMF with two filters, for experiment 4 (faster motion).

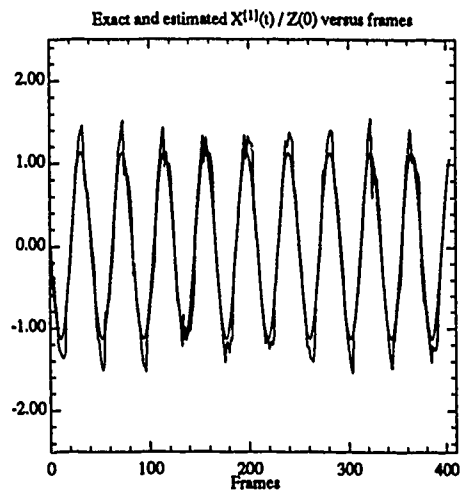


Fig. 8b. Exact and estimated $X^{(1)}(t) / Z(0)$ versus number of frames by using FLAT MMF with five filters, for experiment 4 (faster motion).

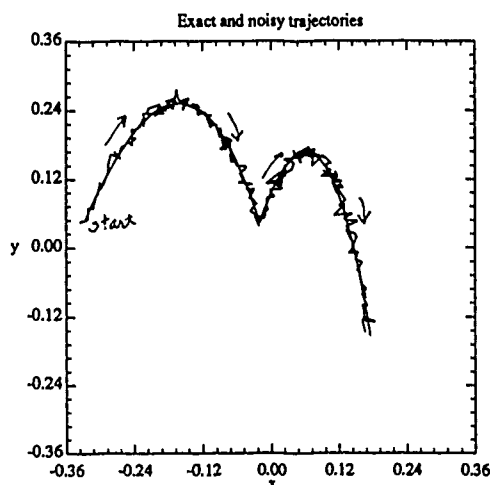


Fig. 9a. Exact and noisy trajectories for experiment 5. Standard deviation of the noise is 2.5 pixels.

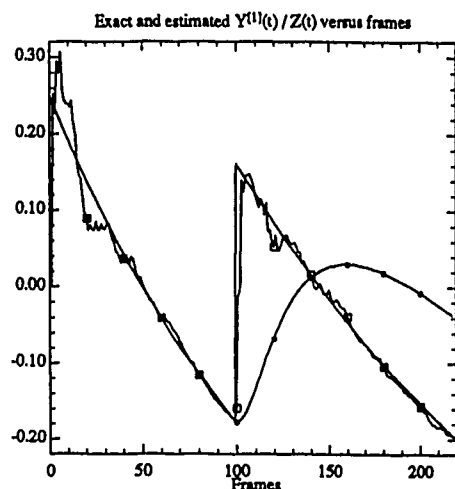


Fig. 9b. Exact and estimated $Y^{(1)}(t) / Z(t)$ versus number of frames by using normal filter (O) and FLAT MMF (□), for experiment 5.

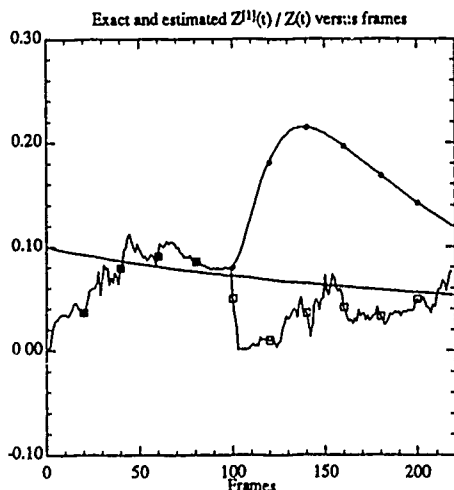


Fig. 9c. Exact and estimated $Z^{(1)}(t) / Z(t)$ versus number of frames by using normal filter (O) and FLAT MMF (□), for experiment 5.

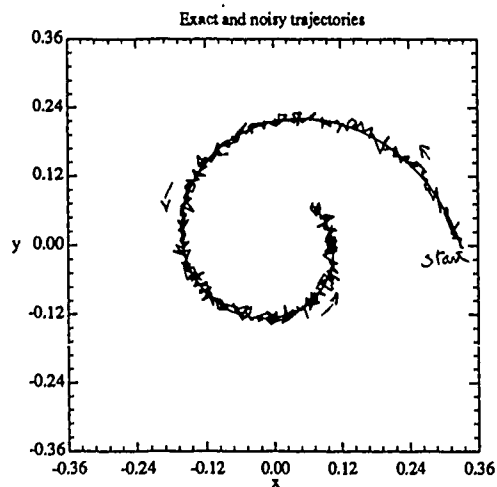


Fig. 10a. Exact and noisy trajectories for experiment 6. Standard deviation of the noise is 2.5 pixels.

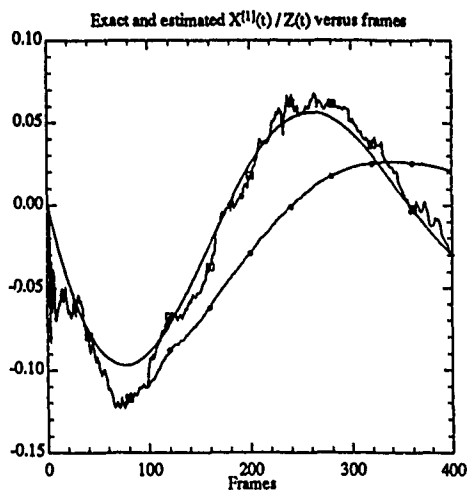


Fig. 10b. Exact and estimated $X^{(1)}(t) / Z(t)$ versus number of frames by using normal filter (O) and FLAT MMF (□), for experiment 6.

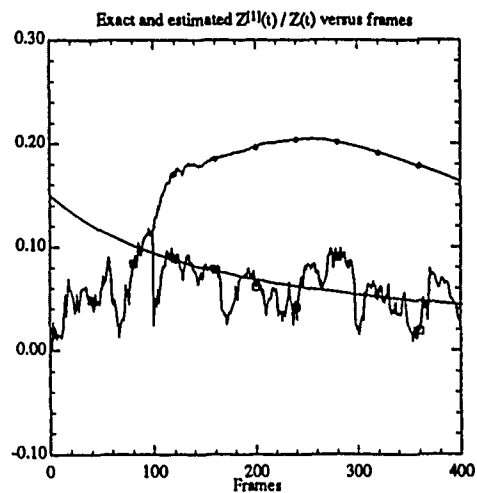


Fig. 10c. Exact and estimated $Z^{(1)}(t) / Z(t)$ versus number of frames by using normal filter (O) and FLAT MMF (□), for experiment 6.

Integrated Multiresolution Image Acquisition and Surface Reconstruction from Active Stereo

SubhODEV Das and NARENDRA AHUJA
Coordinated Science Laboratory and Beckman Institute
University of Illinois at Urbana-Champaign
Urbana, IL 61801

Abstract

This paper is concerned with the problem of surface reconstruction from stereo images for large scenes having large depth ranges, where it is necessary to aim cameras in different directions and to fixate at different objects. In the past we have reported an approach to acquiring multiresolution surface information and using it to devise a strategy for systematic sensing of different parts of the scene. This paper concentrates on the selection of new fixation points from among the nonfixated, low resolution scene parts, and subsequent processing for surface reconstruction. The coarse stereo estimate of the new fixation point is refined as the images of the new fixation point gradually deblur during the process of refixation and is subsequently used to analyze the fixated parts of the scene.

1 Introduction

This paper is concerned with the problem of surface reconstruction from stereo images for large scenes having large depth ranges. At any stage of such a surface reconstruction process, sharp images can be acquired only for narrow parts of the visual field, capturing a limited depth range. The high resolution parts of the scene, contained within the depth of field of the cameras, are said to constitute a *central visual field*, while the low resolution parts out of the depth of field, and typically away from the image center, are said to belong to the *peripheral visual field* [Das and Ahuja, 1989]. Accurate surface map is extracted for the central visual field by integrating the use of camera focus, camera vergence, and stereo disparity [Abbott and Ahuja, 1988]. When the entire surface of the fixated object has been scanned, the acquired surface map does not smoothly extend, and therefore surface reconstruction must be resumed by fixating on a new object, selected from the periphery of the current visual field. This presents a dilemma since the exact locations and shapes of "new objects" are unknown (otherwise there would be no need for fixation and subsequent surface reconstruction.)

This research was supported in part by the National Science Foundation under grant IRI-89-11942, Army Research Office under grant DAAL 03-87-K-0006, and State of Illinois Department of Commerce and Community Affairs under grant 90-103.

We present an approach to using coarse structural information about the scene in selecting a new fixation point in the peripheral field and acquiring structural information in the vicinity of the selected point at increasing resolution as the cameras reconfigure and aim at the point.

Section 2 describes in greater detail the background and motivation behind the work reported in this paper. Section 3 presents an algorithm that interleaves coarse-to-fine acquisition of stereo images with their analysis for coarse-to-fine surface reconstruction. Section 4 gives details of implementation and the experimental results.

2 Background and Motivation

In this section we summarize the past research related to the work reported in this paper, and the motivations that lead to the development of the approach described in the following sections.

2.1 Background

This paper pursues the basic theme of active, intelligent data acquisition [Bajcsy, 1985, Bajcsy, 1988]. Computational active vision has become more feasible in the recent years with the availability of sophisticated hardware for controlling imaging elements [Ballard, 1989, Burt, 1988, Clark and Ferrier, 1988, Krotkov, 1987]. In their analysis of surface reconstruction from stereo images, Marr and Poggio [Marr and Poggio, 1979] also point out the role of eye movements in providing large relative image shifts for matching stereo images having large disparities, thus implying the need for active data acquisition. Ballard and Ozcanarli [Ballard and Ozcanarli, 1988] point out that the incorporation of eye movements radically changes (simplifies) many vision computations; for example, the computation of depth near the point of fixation becomes much easier. Aloimonos et al [Aloimonos et al., 1987] show that active control of imaging parameters leads to simpler formulations of many vision problems that are not well behaved in passive vision. Geiger and Yuille [Geiger and Yuille, 1987] describe a framework for using small vergence changes to help disambiguate stereo correspondences. Abbott and Ahuja [Abbott and Ahuja, 1988] demonstrate the efficacy of integrating image acquisition and image analysis for a single object, by interleaving the processes of camera vergence and focusing with those of depth estimation from camera focus and stereo disparity. Shmuel and Werman [Shmuel and Werman, 1990] have consid-

ered the related problem of surface map generation from multiple viewpoints; they use iterative Kalman-filtering techniques to predict a new camera pose for maximal reduction of uncertainty in depth information. Some recent studies have considered higher level criteria for fixation (called *attention*), e.g., for recognition [Bolle *et al.*, 1990].

2.2 Motivation

Consider the initial state in which one of the objects in a scene is fixated on. Any parts of the scene in the peripheral visual field appear out of focus, with the degree of blur determined by the distance from the fixation point. Stereo analysis of the out of focus peripheral image regions would result in surface estimates which would be inaccurate due to poor localization of features. This creates an ordering on different parts of the scene such that the earlier a part is in the ordering the better is the accuracy of its surface estimate.

Traditionally, the scope of stereo has been restricted to provide accurate depth estimate from sharp images for the parts of the scene corresponding to the beginning of the ordering. However, stereo can also be used to obtain inaccurate estimates for peripheral objects that occur later in the ordering. In fact, the availability of coarse peripheral maps would make it possible to select a new fixation point on a new object. Once the cameras are fixated at the newly selected object, the resolution of the rest of the objects lying in the direction of the selected object also improves. Therefore, as the finest stereo reconstruction is achieved for the selected object, the accuracy of the surface information available for those other objects which are now closer to the fixation point also improves.

The availability of the coarse depth map for the unmapped parts of the scene has advantages other than the ability to select a new fixation point. While moving from one fixation point to the next, the mechanical reconfiguration of the image planes is not instantaneous. Intermediate images are obtained with decreasing blur which may be continuously stereo analyzed to improve the estimate for the new point. In this process, the computational blurring operation is replaced by instantaneous optical blurring. The number of stereo pairs acquired before fixation is achieved would depend on the amount of image plane reconfiguration required. The improved coarse depth estimate from stereo can help in predicting and expediting the search for the best focus axis setting corresponding to the new fixation point and the camera vergence.

The ordering discussed above is defined by the increasing depth property - the closest object is fixated on and stereo analyzed first followed by the next closest object. There are two computational advantages of using such a near-to-far scan. One, by reconstructing the surfaces of the near objects first, the occluded portions of the farther objects can be identified. Thus, knowing those parts of the scene which are occluded from at least one viewpoint would avoid selection of such points for fixation. The second advantage in starting with the near objects is that doing so maximally exploits the focus cue

which is computationally simpler but more effective for short range objects. The exact, global order in which objects are selected for fixation will also depend on their locations in the scene in addition to their depths.

3 Algorithm

In this section we describe an algorithm to achieve the desired integration of multiresolution image acquisition and their coarse-to-fine processing. The algorithm consists of the following steps: (1) For a given fixation point, coarse peripheral surface map is obtained along with fine map for the current central visual field; (2) an unoccluded peripheral point whose selection involves minimum lateral movement of cameras and reconfiguration of their image planes is chosen as the new target point; and (3) a sequence of images of increasing resolution is acquired and stereo analyzed, thus obtaining surface maps with increasing accuracy, during the time the cameras verge and focus on the new target point. These steps of the algorithm are discussed in the following subsections.

3.1 Acquisition of Multiresolution Surface Map

Stereo images are acquired with a focal length (f_{stereo}) smaller than the one used for estimating depth from focus (f_{focus}) to increase the field of view. The fixation point is in focus in these images. The parts of the scene that are in sharp focus (corresponding to objects that lie within the depth of field of the scene) are segmented out [Das and Ahuja, 1989] to constitute the central field of view and the defocused regions comprise the peripheral field. If the point spread function (p.s.f) of a finite aperture lens is modeled by a 2D Gaussian then the spread parameter σ_l of the Gaussian signifies the degree of optical blurring of a defocused point. The parameter σ_l is proportional to the focal length, aperture and the distance of the defocused point from the fixation point.

Stereo reconstruction for the high resolution (using an $N \times N$ grid) central visual field takes place using a small value of σ (σ_{cl}) for the Laplacian of Gaussian ($\nabla^2 G$) feature detector. This σ gives the best trade-off between localization and stability of the detected zero-crossings (features). The stereo estimates from previous fixation, wherever available, are used as initial estimates for the stereo analysis, else the stereo-based estimate of the depth of the current fixation point from target homing (Section 3.3) is used. The result of stereo reconstruction is a high resolution *fine* (accurate) surface map for the central visual field.

Matches for the defocused ($\sigma_l = \sigma_s$) peripheral features in any unmapped parts of the scene (coarse estimates obtained during previous fixation are used for the mapped parts) that are uncovered during camera reconfiguration are found by searching over large image regions. A σ_{pfl} larger than σ_{cl} is used for the peripheral feature detector to introduce additional smoothing so that the number of matchable features be small. The Gaussian expressing the optical and computational blurring effects at a given peripheral point has a spread parameter of $\sigma_t = \sqrt{\sigma_l^2 + \sigma_{\text{pfl}}^2}$. In addition to smoothing, the periphery is subsampled using an $M \times M$ grid ($M < N$). The effects of blurring and subsampling are

detrimental to the accuracy of stereo resulting in a low resolution *coarse* (inaccurate) surface map for the peripheral visual field.

3.2 Selection of a New Target

At this stage, the surface of an object currently in focus has been estimated, and the extension of the surface map must resume by fixating at another object. The availability of the peripheral surface map makes the selection of a new fixation point possible, albeit with limited accuracy, and thus helps to avoid the need for knowing object depths before they are estimated!

Given an approximate surface map in the peripheral visual field, how should we select a fixation point? In [Abbott and Ahuja, 1988] some criteria were identified for selection of a fixation point which were motivated by known characteristics of fixation in human vision as well as computational considerations. We use similar criteria here. A target point at position p , in a coordinate system fixed with respect to the camera locations, is chosen from the current periphery so that the following weighted average is minimized:

$$E = a_1 \|p - p_{CAM}\| + a_2 \|p - p_{POF}\| + a_3 A(p, p_{POF}) \quad (1)$$

where p_{CAM} and p_{POF} denote the locations of camera reference frame and the current point of fixation, respectively; $\|\cdot\|$ is the Euclidean distance norm; and the function A gives the angular separation between two 3D points in the camera reference frame. Candidate target p must be visible to both viewpoints, and must lie within camera travel limits.

The first term enforces a near-to-far ordering on fixation points whose advantages are explained in the previous section. The second term favors selection of an object close to currently fixated object since the closer it is the more accurate the target location information from the peripheral map is. The third term biases the choice of target to scene points which lie in directions close to that of the current fixation point, preventing large angular movements of the cameras between fixations.

3.3 Target Homing

Once a target point has been selected on a new object, the cameras need to be reconfigured to fixate on the point. This involves changing camera orientations and focus settings. Such homing on to the target is attempted using the largest available focal length (f_{focus}) that causes significant blurring ($\sigma_l = \sigma_{lf}$) of the target point.

Initially, the stereo based depth estimate of the peripheral target point is inaccurate due to blurring by a Gaussian kernel of spread parameter σ_l (Section 3.1). The target point which is outside the depth of field of the current focus setting is significantly blurred (optical, $\sigma_l = \sigma_{lf}$). As the image plane is gradually reconfigured by changing the focus settings, stereo images are acquired continuously. Each pair of optically blurred images is subsampled, reducing the degree of subsampling as images become less blurred (σ_{lf} decreases). Let $H_i \times H_i$ denote the resolution of the sampled images at

the i th stage ($\sigma_{lf} = \sigma_{lf}^i$) during reconfiguration:

$$\frac{H_i}{M} = \frac{\sigma_l}{n\sigma_{lf}^i} \quad \text{and} \quad \frac{H_i}{H_{i+1}} = \frac{\sigma_{lf}^{i+1}}{\sigma_{lf}^i} \quad (2)$$

where $f_{focus}/f_{stereo} = n, n > 1$. Since the optically blurred images are obtained continuously, the improvement in the stereo-based depth estimate of the target point from the analysis of two consecutive image pairs is significant only when the difference $\Delta\sigma = \sigma_{lf}^i - \sigma_{lf}^{i+1}$ is significant. Hence, the intermediate images in which the blur of the target point is between σ_{lf}^i and σ_{lf}^{i+1} are skipped for stereo analysis.

Features are detected in the stereo images using Nevatia-Babu operator [Nevatia and Babu, 1980]. The surface estimates derived from an image pair at any stage during camera reconfiguration serve as coarse estimates for surface reconstruction from later images acquired with smaller σ_{lf} . This process of coarse-to-fine image acquisition interleaved with surface reconstruction is continued till σ_{lf} reaches σ_{crit} , at which stage surface reconstruction for the next object is initiated.

3.4 Target Fixation

The target homing stage terminates with the cameras oriented such that the estimated target point location falls at the center of each image. The increasingly improved stereo estimate obtained during target homing brings the two cameras close to focus and sets up the approximate vergence angle for the cameras. In order to focus the cameras exactly, this depth estimate Z_s is used to establish an interval of focus axis settings $[p_0, p_1]$ symmetric about an axis setting p ($p_0 < p$ and $p_1 > p$). This interval which corresponds to the depth of field at p is finely quantized and searched for a peak of the focus criterion function, defined as the total squared gradient over a fixation window centered at the target point. As in [Abbott and Ahuja, 1988] we perturb the camera orientations slightly to maximize sharpness of images and the correlation between the area around the target locations (image centers), which serves as the final step in fixation.

4 Implementation and Results

In this section we present details and results of implementing our active stereo algorithm on a dynamic imaging system. The system consists of two Cohu 4815 CCD cameras mounted on a stereo platform and equipped with Vicon V17.5-105M motorized zoom lenses. High-precision stepper-motor rotational units are used to control independent pan, tilt and vergence angles. The imaging system is controlled by a Sun Microsystems 3/160 workstation.

4.1 Implementation Details

For the left and the right cameras focal lengths (calibrated) of $f_{stereo} = 47.7$ mm and 47.2 mm are used to acquire the stereo images, and $f_{focus} = 105.4$ mm and 101.0 mm (full zoom) are used in the fixation process. The baseline between the cameras is 28 cm. The parameters of (1) are chosen as $a_1 = 0.25$, $a_2 = 0.5$ and

$\alpha_3 = 0.25$; $\sigma_{cfl} = 6$ and $N = 256$ for the central visual field; $\sigma_{pfl} = 9$ and $M = 128$ for the peripheral field; and $\Delta\sigma_{lf} = 3$ is used.

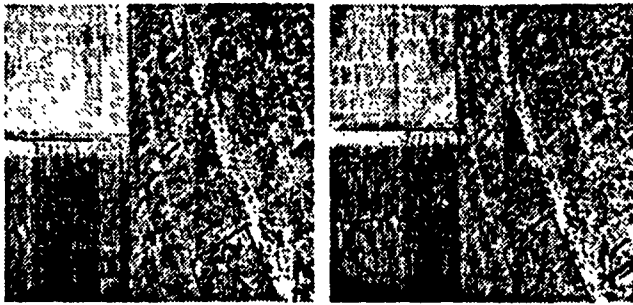
4.2 Experimental Results

The dynamic camera system was made to scan an indoor scene consisting of a vertical barrel (approximately cylindrical) next to a rectangular box, both resting on a flat table top and in front of a rear wall. During one of the fixations of the barrel the stereo images of Figure 1 are acquired. Here, the barrel being in focus occupies the central visual field while the box and the back wall constitute the peripheral visual field. The fine central range map together with the coarse peripheral range map for this fixation is shown for the left viewpoint in Figure 2. A window in Figure 3 marks the newly selected target point on the box which minimizes (1). The world coordinates of the new target from the coarse stereo depth estimate are (0.308, 0.098, 2.188), all in meters.

Upon selecting the target the system aims the cameras at it. The focal length of each camera is set to full zoom as required by the fixation process resulting in the optically blurred left and right images of Figure 4, $\sigma_{lf}^1 = 8$. During the previous fixation $\sigma_{ls} = 4$ for the new target point and $\sigma_{pfl} = 9$, hence $\sigma_t = \sqrt{\sigma_{ls}^2 + \sigma_{pfl}^2} = 9.9$. Using these values of σ_{lf}^1 and σ_t in (2) and taking the $\lceil n\sigma_{lf}^1/\sigma_t \rceil$, $H_1 = 64$. Nevatia-Babu line extraction algorithm is used to detect features which are matched to obtain the coarse stereo map of Figure 5. The recomputed depth the target from stereo is 2.228 m. The next set of images that would have been stereo analyzed would have $\sigma_{lf}^2 = \sigma_{lf}^1 - \Delta\sigma_{lf} = 5$. But $\sigma_{lf}^2 < \sigma_{cfl}$, and the mechanical reconfiguration is therefore continued without stereo analysis until the focus setting corresponding to the depth of 2.228 m has been attained. To fixate the target, the search interval of focus axis settings is $p_0 = 5355$ and $p_1 = 5714$ (left camera) and $p_0 = 5282$ and $p_1 = 5687$ (right camera). The peak of the focus criterion function is detected at $p_l = 5578$ (left camera) and $p_r = 5675$ (right camera). The focus based depth estimate is 2.252 m while the measured distance is 2.18 m. The stereo images of Figure 6 have the box occupying the central visual field while the barrel and the wall belong to the peripheral field with the barrel being less peripheral (blurred) than the wall. The coarse map for the box is now replaced with a fine map as shown in Figure 7 which has been added to the composite map in Figure 8 that previously contained only estimates for the barrel.

References

- [Abbott and Ahuja, 1988] A. Lynn Abbott and Narendra Ahuja. Surface reconstruction by dynamic integration of focus, camera vergence, and stereo. In *Proc. Second Intl. Conf. on Computer Vision*, pages 532-543, Tarpon Springs, FL, December 1988.
- [Aloimonos et al., 1987] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. In *Proc. First Intl. Conf. on Computer Vision*, pages 35-54, London, UK, June 1987.
- [Bajcsy, 1985] Ruzena Bajcsy. Active perception vs. passive perception. In *Proc. Workshop on Computer Vision*, pages 55-59, Bellaire, MI, October 1985.
- [Bajcsy, 1988] Ruzena Bajcsy. Perception with feedback. In *Proc. DARPA Image Understanding Workshop*, pages 279-288, Cambridge, MA, April 1988.
- [Ballard and Ozcandarli, 1988] Dana H. Ballard and A. Ozcandarli. Eye fixation and early vision: Kinetic depth. In *Proc. Second Intl. Conf. on Computer Vision*, pages 524-531, Tarpon Springs, FL, December 1988.
- [Ballard, 1989] Dana H. Ballard. Reference frames for animate vision. In *Proc. 11th IJCAI*, pages 1635-1641, Detroit, MI, August 1989.
- [Bolle et al., 1990] Rudd M. Bolle, Andrea Califano, and Rick Kjeldsen. Data and model driven focus of attention. In *Proc. 10th Intl. Conf. on Pattern Recognition*, pages 1-7, Atlantic City, NJ, June 1990.
- [Burt, 1988] Peter J. Burt. Algorithms and architectures for smart sensing. In *Proc. DARPA Image Understanding Workshop*, pages 139-153, Cambridge, MA, April 1988.
- [Clark and Ferrier, 1988] J. J. Clark and N. J. Ferrier. Modal control of an attentive vision system. In *Proc. Second Intl. Conf. on Computer Vision*, pages 514-523, Tarpon Springs, FL, December 1988.
- [Das and Ahuja, 1989] Subhdev Das and Narendra Ahuja. Integrating multiresolution image acquisition and coarse-to-fine surface reconstruction from stereo. In *Proc. IEEE Workshop on Interpretation of 3D Scenes*, pages 9-15, Austin, Texas, November 1989.
- [Geiger and Yuille, 1987] Davi Geiger and Alan Yuille. Stereopsis and eye-movement. In *Proc. First Intl. Conf. on Computer Vision*, pages 306-314, London, UK, June 1987.
- [Krotkov, 1987] Eric P. Krotkov. Exploratory visual sensing for determining spatial layout with an agile stereo camera system. Ph.D. Thesis MS-CIS-87-29, GRASP Laboratory, University of Pennsylvania, Philadelphia, PA, 1987.
- [Marr and Poggio, 1979] David Marr and Tomaso Poggio. A computational theory of human stereo vision. In *Proc. the Royal Soc. of London, vol. B, no. 204*, pages 301-328, 1979.
- [Nevatia and Babu, 1980] Ramakant Nevatia and K. R. Babu. Linear feature extraction and description. *Computer Graphics and Image Processing*, 13:257-269, 1980.
- [Shmuel and Werman, 1990] Amir Shmuel and Michael Werman. Active vision: 3d from an image sequence. In *Proc. 10th Intl. Conf. on Pattern Recognition*, pages 48-54, Atlantic City, NJ, June 1990.



(a)

(b)

Figure 1: Stereo image pair, (a) left and (b) right, after the fixation of the barrel.

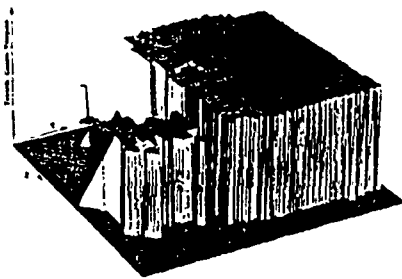


Figure 2: High resolution central range map of the barrel and coarser peripheral range map of the box.

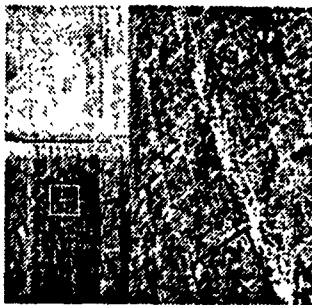
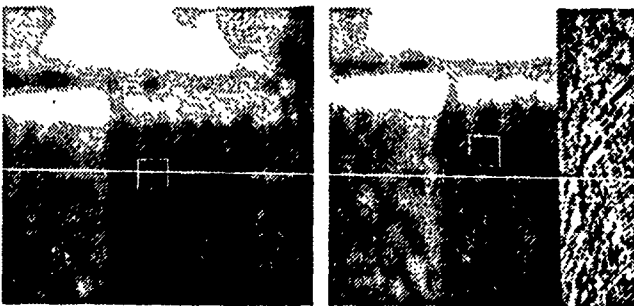


Figure 3: A window marks the new target (on the box) chosen by minimizing the target selection criterion.



(a)

(b)

Figure 4: Coarse (a) left and (b) right images at full zoom as the cameras begin homing on the new target.

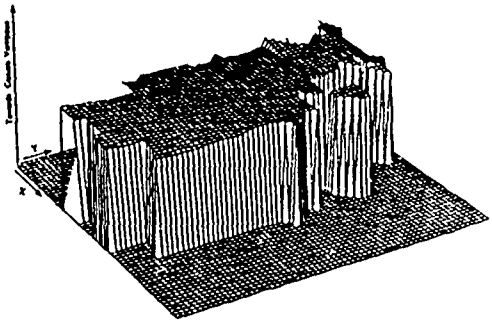


Figure 5: The coarse stereo map in the vicinity of the target point from the optically blurred images (64×64).



(a)

(b)

Figure 6: Stereo image pair, (a) left and (b) right, with the box in the central visual field while the back wall continues to occupy the peripheral field.

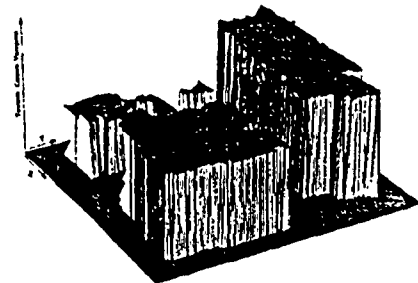


Figure 7: A higher resolution (than in Figure 2) range map for the box along with a coarse peripheral map for the wall.

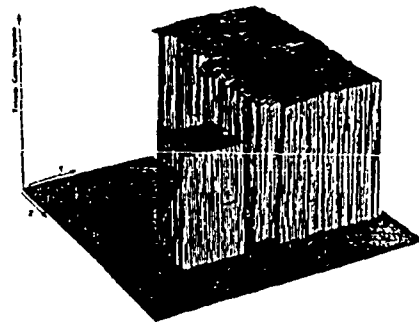


Figure 8: The composite range map is updated by adding the reconstructed surface of the box.

A Dynamical Systems Approach to Integration in Stereo

Edward J. Altman and Narendra Ahuja
University of Illinois at Urbana-Champaign
Beckman Institute
Urbana, IL 61801 USA

Abstract

An analog approach to stereo integration of feature detection, matching, and surface interpolation is presented. The integration is accomplished through the monolithic use of *dynamical systems* as the computational mechanism. The presence of different features in an image neighborhood is indicated by the normal mode energies of a *feature detection dynamical system* when initialized by the gray-levels in the neighborhood. The system converges to one of a set of n stable states, thereby detecting one of n possible features. Stereo disparity of a feature is computed by a *disparity dynamical system* which identifies that candidate as a match whose feature dynamical system contains the largest normal mode energy among the candidates. The smoothness of disparity is enforced by introducing *spatial coupling* among disparity dynamical systems. Simulation results for a one-dimensional implementation are given.

1. Introduction

The traditional formulation of the problem of estimating three-dimensional surfaces from stereo images consists of three steps: feature detection, feature matching, and surface interpolation. It has been argued that the latter two tasks are more accurately executed in an integrated manner rather than sequentially since they are strongly interdependent [Hoff and Ahuja, 1985, Boulton and Chen, 1988, Eastman and Waxman, 1987]. For example, when an image neighborhood is relatively featureless, or there are multiple matches possible for a feature, matching may lead to ambiguous results. Alternately, noise may change image gray levels resulting in none or wrong matches for a point. In such circumstances, matching of a point may be performed more accurately by taking into account the matching decisions in the vicinity of the point. We have described a digital realization of integration in which the integration of matching and surface interpolation tasks is approximated by their interleaving, and iteration of the interleaved sequence

at increasingly fine image resolutions down a coarse-to-fine representation hierarchy [Hoff and Ahuja, 1989]. The integration results in a number of advantages not possible to obtain otherwise [Hoff and Ahuja, 1989].

This paper is primarily motivated by the desire to achieve (i) a more *complete* integration, than realized using our above-referred, macrolevel, temporal interleaving scheme, and (ii) integration of all *three* steps, instead of just the latter two. To this end, we present an analog formulation which implements the three stages of stereo using the same analog process, as well as the integration of feature detection, feature matching and surface estimation.

Section 2 presents an overview of our formulation. Section 3 reviews basic concepts from dynamical systems used in this paper. Section 4 discusses the feature dynamical system. Section 5 briefly describes the disparity dynamical system and Section 6 summarizes the modifications of the disparity dynamical system needed to enforce surface smoothness. For brevity, we are not able to present the analytical details from the dynamical systems theory which we have used in developing the formulation presented in this paper. Simulation results are provided in Sections 4-5. Section 7 presents concluding remarks.

2. Overview

Before presenting the details, we will first explain the concept of a nonlinear dynamical system which we have used to develop the formulation of integration presented in this paper. A nonlinear dynamical system models the temporal evolution for a set of variables and their interrelationships through a system of differential equations. In this paper, each variable is viewed as representing a nonlinear oscillator. The variables exhibit mutual coupling (interdependence) and change nonlinearly with time. The values taken by the variables and interrelationships among them define states of the system. A dynamical system which is stable, converges to one of a set of what are called *stable states*, or *normal modes*, of that system for any set of initial values of the variables. On convergence, the oscillatory energy of the state variables of the dynamical system is concentrated in one normal mode with a common frequency for all variables. The "computation" performed by the system is represented by the specific normal mode to which the system converges.

* This research was supported in part by the National Science Foundation under grant IRI-89-11942, Army Research Office under grant DAAL 03-87-K-0006, and State of Illinois Department of Commerce and Community Affairs under grant 90-103.

The incorporation of such dynamics into the formulation of binocular stereopsis brings in new complexities, since it introduces a time factor into stereo analysis which is otherwise only spatial. However, the temporal behavior adds a new and common dimension for formal interaction and integration among the different types of operations that comprise the three steps for stereo. The characteristics of temporal behavior, as exemplified by the stable oscillatory modes, can now serve as the common language for cooperation among the different information sources being integrated.

The formulation in this paper uses two types of dynamical systems. The first, called the *feature dynamical system*, is concerned with feature detection in an image neighborhood. Simple features, such as vertical or horizontal edges, can be described by relations among the gray-level values of pixels in an image neighborhood. The co-occurrence of a set of gray-levels satisfying these relations indicates the presence of the feature in the image. The feature dynamical system (1) detects features from the co-occurrence of such pixel values. The convergence of the system to one of its normal modes of oscillation achieves the analog classification of the image neighborhood. The amount of energy present in each normal mode of the dynamical system provides a measure for the presence of the corresponding feature.

A second type of system, called the *disparity dynamical system* integrates feature matching and disparity detection. This system is a winner-take-all network which converges to a stable state based upon the binocular co-occurrence of similar features. The inputs in this case are obtained from the distribution of normal mode energies along an epipolar line composed of k feature dynamical systems. There is one variable d_i for each feature dynamical system, thus there are as many variables as the number of discrete disparity values possible. The stable state with the largest d_i indicates the disparity value. Since each feature dynamical system has n normal modes, the disparity dynamical system selects as its set of inputs from the epipolar line that mode which is the same as the dominant mode in the feature dynamical system at its own location, thus realizing the task of feature matching.

The convergence rate of each of the above dynamical systems is proportional to the ambiguity of the classification of the initial state. Consider, for example, the feature dynamical system. In our current implementation, this system has four stable states corresponding to a horizontal edge, vertical edge, diagonal edge, or a uniform region. If the gray-level distribution of a given neighborhood of pixels does not resemble any of these structures, i.e., there is a low level of similarity between the neighborhood structure and the closest stable state, the dynamical system will take a long time to converge, and thus be more susceptible to interactions from other dynamical systems. As another example, consider the disparity dynamical system and the case where there is no match along the epipolar line. Then all energy inputs to the dynamical system will be small and the convergence will be slow. Finally, if there are multiple

comparably good matches, then competition among the stable states corresponding to each of the good matches will drive the system into an intermediate state which can be strongly influenced by interactions from nearby systems.

Interestingly, as mentioned at the beginning of this section, the above, ambiguous initial states correspond precisely to the situations that have motivated integration in the previous work, wherein information from a pixel's vicinity is propagated to the pixel to disambiguate or override the local decisions. To incorporate such spatial interaction, we modify our disparity dynamical system so that the stable state it converges to is influenced by the disparity variables at nearby locations. Thus, the disparity variables are not only constrained by the matching of features, but also by the disparity variables at nearby locations. Whenever ambiguities arise, the corresponding dynamical systems are slow to converge; convergence towards any such stable states which are in common with nearby dynamical systems associated with unambiguous neighborhoods is then accelerated due to local support from these systems.

3. Dynamical Systems Concepts

A *dynamical system* is a system of differential equations where the relations among the state variables can be used to represent relationships among physical quantities. When the values for the state variables of a nonlinear dynamical system are confined to a closed surface, that surface is called an *integral manifold* which is a nonlinear generalization of the notion of invariant subspaces in linear systems theory [Hale, 1980].

The complex interdependencies among the system variables can often be simplified by transforming the dynamical system to *normal form* [Guckenheimer and Holmes, 1983]. The normal form of a nonlinear dynamical system is analogous to the representation of a linear system in terms of an orthogonal basis set. The *normal modes* of a dynamical system are the oscillatory states of the normal form system where each variable has a common frequency. The *energy* of a normal mode is a measure of the magnitude of oscillations for the state variables.

4. Feature Dynamical System

Let the set $\{x_i\}$ denote the state variables of a dynamical system. The gray-level values of a set of pixels in an image neighborhood are sampled at time t_0 to establish the initial values of the corresponding x_i . The specific feature dynamical system we use in our current implementation is an ensemble of four coupled van der Pol oscillators

$$\begin{aligned}\ddot{x}_1 - \epsilon(1 - x_1^2)\dot{x}_1 + \omega_0^2 x_1 &= \alpha(-x_1 + x_2) \\ \ddot{x}_2 - \epsilon(1 - x_2^2)\dot{x}_2 + \omega_0^2 x_2 &= \alpha(x_1 - x_2 + x_3) \\ \ddot{x}_3 - \epsilon(1 - x_3^2)\dot{x}_3 + \omega_0^2 x_3 &= \alpha(x_2 - x_3 + x_4) \\ \ddot{x}_4 - \epsilon(1 - x_4^2)\dot{x}_4 + \omega_0^2 x_4 &= \alpha(x_3 - x_4)\end{aligned}\quad (1)$$

where

- x_i — state variables for the dynamical system,
- ω_0 — natural frequency of each isolated oscillator,
- α — coupling strength within an ensemble,
- ϵ — strength of nonlinearity in the system.

and the terms on the right-hand side indicate the coupling of oscillators within the ensemble. The initial state for each variable is the gray-level associated with an image pixel. Using normal form theory, it can be shown that a set of n possible features can be identified through the convergence of the dynamical system (1) to one of n stable modes of coherent oscillations corresponding to the normal modes of the feature dynamical system. The actual features are detected by observing the energies in the normal mode oscillations of the dynamical system, each of which serves as a measure for the presence of a specific feature.

The matrix form of (1) is expressed by

$$\ddot{\mathbf{x}} + \mathbf{B}_n \mathbf{x} = \epsilon \left(\dot{\mathbf{x}} - \frac{1}{3} \dot{\mathbf{x}}_c \right) \quad (2)$$

where $\mathbf{x} = [x_1, \dots, x_n]$, $\mathbf{x}_c = [x_1^3, \dots, x_n^3]^T$, the term on the right represents the nonlinearity of the system, and

$$\mathbf{B}_n = \begin{bmatrix} 1 + \alpha & -\alpha & 0 & 0 \\ -\alpha & 1 + \alpha & -\alpha & 0 \\ 0 & -\alpha & 1 + \alpha & -\alpha \\ 0 & 0 & -\alpha & 1 + \alpha \end{bmatrix} \quad (3)$$

is the coupling for $n = 4$. The frequencies of the oscillatory modes are determined by the eigenvalues of \mathbf{B}_n .

The dynamical system (1) can also be written as a system of first order equations

$$\begin{aligned} \dot{x}_1 &= \epsilon \left(x_1 - \frac{1}{3} x_1^3 \right) - \omega_0 y_1 & \dot{y}_1 &= \omega_0 x_1 - \frac{\alpha}{\omega_0} (x_2 - x_1) \\ \dot{x}_2 &= \epsilon \left(x_2 - \frac{1}{3} x_2^3 \right) - \omega_0 y_2 & \dot{y}_2 &= \omega_0 x_2 - \frac{\alpha}{\omega_0} (x_1 - x_2 + x_3) \\ \dot{x}_3 &= \epsilon \left(x_3 - \frac{1}{3} x_3^3 \right) - \omega_0 y_3 & \dot{y}_3 &= \omega_0 x_3 - \frac{\alpha}{\omega_0} (x_2 - x_3 + x_4) \\ \dot{x}_4 &= \epsilon \left(x_4 - \frac{1}{3} x_4^3 \right) - \omega_0 y_4 & \dot{y}_4 &= \omega_0 x_4 - \frac{\alpha}{\omega_0} (x_3 - x_4) \end{aligned}$$

where the x_i 's are initialized by the image pixels and the y_i 's are initialized to zero. Experimental results for this system are illustrated in Figure 1 where inputs from different 2x2 gray-level patterns lead to the different stable modes of oscillation among the state variables $[x_1, \dots, x_4]$.

Linearization of the above system by the method of normal forms results in an equivalent dynamical system with four oscillatory modes. A system with two oscillatory modes has been investigated extensively by [Guckenheimer and Holmes, 1983] using the methods of normal forms and bifurcation theory. Analysis for a system with n coupled oscillators has been reported in [Chua and Endo, 1988] using the method of integral manifolds. Interactions among dynamical systems have been investigated in [Altman, 1990] using integral manifolds of the normal form equations.

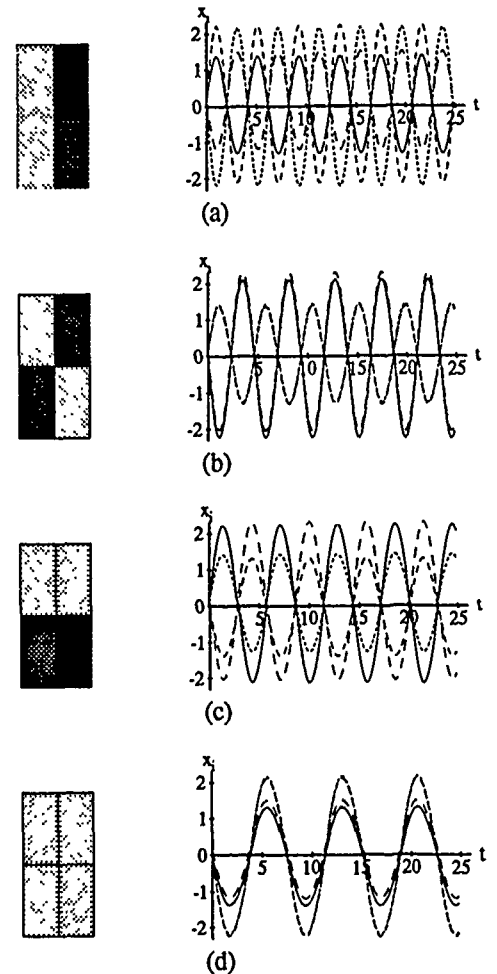


Figure 1: Simulation results for stable oscillatory modes of the feature dynamical system (1): (a) 2x2 vertical edge pattern and the coherent oscillations among the state variables $[x_1, \dots, x_4]$, (b) diagonal edge, (c) horizontal edge, and (d) homogeneous region. The parameter values used; $\epsilon = 0.1$, $\alpha = 0.5$, and $\omega_0 = 1$.

4.1. Normal form of the dynamical system

The dynamical system exhibits complex interdependencies among the system variables which may be simplified by construction of the normal form. The normal form is obtained by first diagonalizing the system (1) using the singular value decomposition of the coupling matrix $\mathbf{B}_n = \mathbf{u} \mathbf{\Lambda} \mathbf{v}^T$ to obtain the coordinate transformation $\mathbf{v} = \mathbf{u} \mathbf{x}$. Energy in the n normal mode oscillations is then obtained by applying the transformation $\rho_i^2 = v_i^2 + \left(\frac{\dot{v}_i}{\omega_0} \right)^2$ and the phase of the oscillations is $\theta_i = \arctan \left(\frac{\dot{v}_i}{v_i} \right)$ for $i \in [1, \dots, 4]$. The

equations describing the normal form of (1) are

$$\begin{aligned}\dot{\rho}_1 &= \frac{\epsilon}{2}\rho_1 - \frac{3\epsilon}{80}\rho_1^3 - \frac{\epsilon}{40}\rho_1(2\rho_2^2 + 2\rho_3^2 + 3\rho_4^2) \\ \dot{\rho}_2 &= \frac{\epsilon}{2}\rho_2 - \frac{3\epsilon}{80}\rho_2^3 - \frac{\epsilon}{40}\rho_2(2\rho_1^2 + 3\rho_3^2 + 2\rho_4^2) \\ \dot{\rho}_3 &= \frac{\epsilon}{2}\rho_3 - \frac{3\epsilon}{80}\rho_3^3 - \frac{\epsilon}{40}\rho_3(2\rho_1^2 + 3\rho_2^2 + 2\rho_4^2) \\ \dot{\rho}_4 &= \frac{\epsilon}{2}\rho_4 - \frac{3\epsilon}{80}\rho_4^3 - \frac{\epsilon}{40}\rho_4(3\rho_1^2 + 2\rho_2^2 + 2\rho_3^2)\end{aligned}\quad (4)$$

Stability analysis shows that this system exhibits four stable states each defined by deterministic phase relations among the four oscillators for which $\rho_i = 2\sqrt{\frac{10}{3}}$ and $\rho_l \approx 0$ for $l \neq i$ and $i \in [1, \dots, 4]$. The stable states of the normal form are related to the stable modes of the original system through a linear transformation $\mathbf{v} = \mathbf{u}\mathbf{x}$. The normal form analysis is used here to obtain the linear coordinate transformation which yields the energy for each of the normal modes. The significance of the normal form analysis is to show that the coherent oscillations among the variables x_i in the feature dynamical system (1) correspond to the normal modes of (4).

The stable states for two of the amplitude equations in (4) are illustrated in Figure 2. The points A and B on the ρ_1 and ρ_2 axes, respectively, are stable fixed points of the amplitude equations and correspond to the energy in two of the stable oscillatory modes of the original system (1). Point D is unstable and point C has a saddle type stability. The segment \overline{CD} forms a *separatrix* between the two stable points at A and B. For illustrative purposes, only two components ρ_1 and ρ_2 have been shown in Figure 2. Similar relationships exist among the other normal modes of the dynamical system. This shows that (4) exhibits the properties of a winner-take-all network.

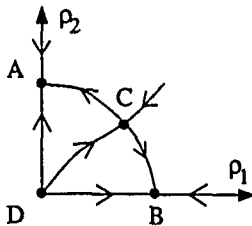


Figure 2: Locations of stable equilibria at A and B for two components of the normal form equation.

5. Disparity Dynamical System

Inputs to the disparity dynamical system are normal mode energies; therefore, we use a dynamical system in which the state variables correspond to energy. The design of the system begins with a winner-take-all network similar to the normal form system (4) where the normal modes now correspond to a discrete set of disparity ranges, rather than feature types.

The detection of k discrete values of disparity, d_i , for $i \in [1, \dots, n]$, can be accomplished by the disparity dynamical

system

$$\dot{d}_i = \mu_i d_i + \lambda_i d_i^3 + \eta_i f_i(\mathbf{d}) \quad (5)$$

where the input to d_i is the normal mode energy of the k th feature of the i th feature dynamical system along an epipolar line. The parameters μ_i , λ_i , and η_i determine the stable states and the convergence properties of the system. The function $f_i(\mathbf{d})$ describes the interactions among the state variables $\mathbf{d} = [d_1, \dots, d_n]$. The task of feature matching is achieved by using the dominant mode of the feature dynamical system at the same location as the disparity dynamical system to select only those inputs which are due to the same normal mode.

5.1. Detection of binocular disparity

The process of disparity detection is illustrated in Figure 3 for 6x6 regions in a stereo pair of images. The epipolar line in the right image is indicated by the ellipse. Two features are present within the ellipse; a vertical edge and a diagonal edge. The distribution of normal mode energies for the feature dynamical systems associated with the epipolar line are displayed in the center of Figure 3(a) where each row shows the energy in the stable state associated with the feature at the left. The disparity dynamical system at the location $L_{0,0}$ is identified by the circle in the left image. Since the feature at $L_{0,0}$ is a diagonal edge, the inputs to the disparity dynamical system are selected from the stable states corresponding to a diagonal edge along an epipolar line.

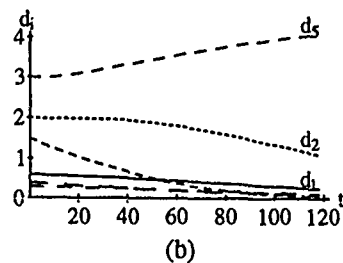
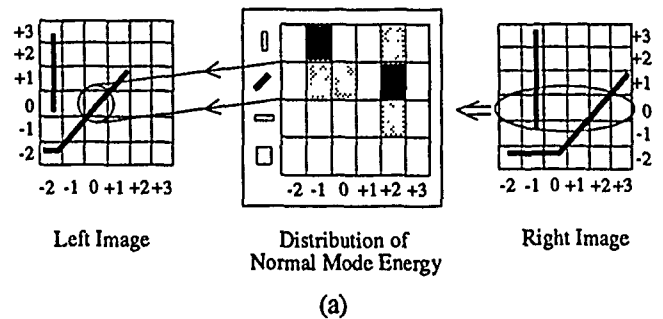


Figure 3: Simulation results for disparity detection using Equation (5) with $n = 6$: (a) stereo pair of images and the distribution of normal mode energies along an epipolar line for the right image (darker shading indicates larger energy), and (b) the selection of a stable state of the disparity dynamical system corresponding to a disparity of +2 units.

Once the feature dynamical systems have converged to the stable states, the state variables d_i for $i \in [1, \dots, n]$ are initialized to the normal mode energies for the diagonal edge. The variable d_i with the highest initial value, (i.e., the diagonal edge with the strongest support among neighboring locations), determines the stable state of the disparity system. There is a 1-to-1 mapping between stable states and disparity ranges

$$d_1 \rightarrow -2 \text{ units}, \dots, d_3 \rightarrow 0 \text{ units}, \dots, d_6 \rightarrow +3 \text{ units} \quad (6)$$

The selection of d_5 as the dominant state variable in Figure 3(b) indicates a disparity value of +2 units.

6. Disparity Dynamical System with Smoothness

Enforcement of surface smoothness is accomplished through spatial coupling of disparity dynamical systems to insure local consistency among the disparity values. To achieve this coupling, we introduce additional variables in the disparity dynamical system (5) corresponding to the normal mode energies of the dynamical systems at adjacent locations in the image. For example, if $d_i^{L_n}$ denotes the energy in the i th mode of the disparity dynamical system at the location L_n in the left image, then the disparity dynamical system at the location L_0 can include spatial interactions

$$\dot{d}_i = \mu_i d_i + \lambda_i d_i^3 + \eta_i f_i(d) + \sum g(d_i^{L_n}) \quad (7)$$

where $g(\cdot)$ describes the spatial interactions. Since the function $f_i(d)$ in (7) generally contains cubic and higher order terms, it is more convenient to convert this system to an equivalent system of coupled oscillators. Normal form theory guarantees that such a transformation always exists [Guckenheimer and Holmes, 1983]. Using this transformation with 1-D coupling, the ensemble of coupled oscillators is given by

$$\ddot{x}_{i,j} - \epsilon(1 - x_{i,j}^2)\dot{x}_{i,j} + \omega_0^2 x_{i,j} = \alpha(x_{i-1,j} - x_{i,j} + x_{i+1,j}) + \beta(x_{i,j-1} + x_{i,j+1}) \quad (8)$$

where the $x_{i,j}$ are a linear combinations of the d_i , j is the positional index, and β is the strength of the spatial coupling. Disparity in this system is now represented by the oscillatory modes rather than the values of the d_i . The advantages of this transformation are: 1) activity is distributed among all components, and 2) most higher order terms are eliminated. It must be noted that this disparity dynamical system enforces local constancy of disparity rather than local smoothness [Hoff and Ahuja, 1989]. Further work is necessary to modify this system so it enforces surface smoothness.

7. Concluding Remarks

The primary advantage of using dynamical systems for feature detection is that multiple tasks can be integrated into

the same analog process. The dynamical system for feature detection integrates the effects of multiple filters or masks commonly used for pattern classification. Dynamical systems also provide a natural method for the integration feature detection, feature matching, and surface interpolation in binocular stereopsis. The inherent parallelism of analog systems matches the parallel nature of stereopsis, and the selection of stable states of the dynamical systems matches the tasks of feature and disparity detection. Finally, the synchronization of systems with oscillatory dynamics matches the grouping of simple objects with similar properties (e.g., the selection of compatible disparities at nearby locations).

The difficulty of designing analog devices provides a primary limitation of this approach. The nonlinear nature of the dynamical systems introduces further complexity into the design which has been reduced by using weakly nonlinear systems. The generalization to strongly nonlinear systems requires careful investigation. The extension of dynamical systems concepts to higher levels of vision and to the integration of multiple sources may require the use of higher order dynamical systems which would greatly increase the complexity of the design.

References

- [Altman, 1990] E. J. Altman. *Dynamical Systems Approach to Binocular Stereopsis*. PhD thesis, University of Illinois, 1990.
- [Boult and Chen, 1988] T. E. Boult and L. H. Chen. Analysis of two new stereo algorithms. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 177-182, Ann Arbor, MI, June 1988.
- [Chua and Endo, 1988] Leon Chua and Tetsuro Endo. Multimode oscillator analysis via integral manifolds part-i: non-resonant case. *Int. J. Circuit Theory Appl.*, 16:25-58, 1988.
- [Eastman and Waxman, 1987] R. D. Eastman and A. M. Waxman. Using disparity functionals for stereo correspondence and surface reconstruction. *Comput. Vision, Graphics, Image Processing*, 39:73-101, 1987.
- [Guckenheimer and Holmes, 1983] J. Guckenheimer and P. Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*. Springer-Verlag, New York, 1983.
- [Hale, 1980] Jack K. Hale. *Ordinary Differential Equations*. R. Krieger, New York, 1980.
- [Hoff and Ahuja, 1985] W. A. Hoff and N. Ahuja. Surfaces from stereo. In *Proc. DARPA Image Understanding Workshop*, pages 98-106, Miami Beach, FL, December 1985.
- [Hoff and Ahuja, 1989] W. A. Hoff and N. Ahuja. Surfaces from stereo: Integrating feature matching disparity estimation, and contour detection. *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI-11(2):121-136, 1989.

Experimental Results of 3D Motion Estimation Using Images of Outdoor Scenes

M. K. Leung, Y. C. Liu and T. S. Huang

Coordinated Science Laboratory
and
Beckman Institute
University of Illinois at Urbana-Champaign
405 North Mathews Avenue
Urbana, IL 61801

Abstract

The main goal of this research is to test how well existing feature extraction/matching and motion estimation algorithms (with appropriate modifications) work on outdoor scenes. For this purpose, a carefully calibrated image sequence data base has been created. The data used for the results reported in this paper consists of a sequence of 24 stereo images of an outdoor scene containing a moving truck with a stationary background. Two motion estimation methods using feature correspondences were applied to the data: Point correspondences over two stereo image pairs, and line correspondences over three monocular images.

Motion estimation methods using feature correspondences consist of two steps: (i) Extract and match features over several images. (ii) Determine the motion parameters from feature correspondences. Both these steps are addressed in this paper. We use algorithms available in the open literature with appropriate (and by no means minor) modifications. Since the emphasis of this paper is on experimental results, we shall not dwell on the details of these modifications.

The image data we use are difficult in the sense that the range to baseline ratio ($\approx 10 : 1$) and the range to focal length ratio ($\approx 900 : 1$) are both large. The question we have tried to answer is: Could existing algorithms with modifications give reasonable motion estimation results? As we shall see, the answer is a qualified yes.

1 Introduction

Motion estimation has evolved into a major research area in computer vision during the last decade. Many algorithms have been proposed. However, most algorithm have been applied to only computer synthesized data, a few to images of indoor scenes, and very few to images of outdoor scenes. And among the last category, we have not found results which are based on data obtained with careful camera calibration and which are compared with carefully calculated ground-truth.

To remedy this situation, we have created, jointly with the U. S. Army Engineer Topographic Laboratory (AI Center) and Purdue University (Photogrammetry Group), a set of image data base which are well calibrated and with ground-truth. The data base consists of stereo image sequences of an outdoor scene containing moving vehicles with a stationary background. The purpose of the present paper is to give some experimental results of applying several motion estimation algorithms to an image sequence from this data base. There is only one moving vehicle, a truck, in the scene, and we aim to estimate its motion in 3-D.

We restrict ourselves to algorithms which are based on only two or three time instants, and which use feature correspondences. In Section 2, we present results using point correspondences over a stereo image sequence, in Section 3, results using line correspondences over a monocular image sequence.

2 Motion Estimation Using Point

Correspondences over a Stereo Image sequence

2.1 Image Data Base

The experiment reported in this paper was performed on a sequence of 24 stereo image pairs taken at consecutive time instants of an U. S. Army 10-ton truck in a parking lot, asphalt surfaced, with trees and one building as the background [1]. The path of the truck was approximately a circular arc. The images were digitized to 4096×4096 pixels (pixel size = $13\mu m$) using a PDS Microdensitometer at U. S. Army Engineer Topographic Laboratories. However, for the experimental work reported in this paper, the original digitized images were sub-sampled to 2048×2048 pixels. Then, a region of 512×512 pixels around the truck on each sub-sampled image was segmented out and used as the input for the experimental work.

2.2 Motion Estimation Using 3-D Point Correspondences

In this section, we present the process of motion estimation using 3-D point correspondences [2]. The process consists of four stages: (1) Determination of point correspondences on two stereo image pairs, (2) Correction of distortions in image coordinates, (3) Derivation of 3-D point coordinates from 2-D correspondences, and (4) Estimation of motion parameters based on the 3-D point correspondences. We shall give a brief description of each stage.

In the first stage, we employ the 4-way matching algorithm suggested in [3] to obtain matched point pairs in two stereo image pairs at two consecutive time instant (t_i and t_{i+1}).

This work was supported by the National Science Foundation Grant IRI-89-08255, the Joint Service Electronics Program Grant N00014-90-J-1270 and State of Illinois Department of Commerce and Community Affairs Grant 90-103.

The algorithm has two steps which are extracting features and matching. The features used in this algorithm are edge points that extracted by locating zero crossings of an image. In the matching step, there are three procedures which are i) stereo matching, ii) time matching and iii) elimination of multiple matches.

Since the image sequence used in this paper was obtained from optical cameras, the images were corrupted by two major distortions of film and lens. The distortion of film is caused by the unstable nature of the acetate base of the commercial film while the distortion of lens is due to aberrations. These two distortions on the image coordinates have to be corrected before any process, such as derivation of 3-D coordinates, can be applied to the matched points. Therefore, after the matching process, the next stage is correction of distortions in the image coordinates. We use the method described in [1, 2] to perform the required corrections. The method consists of two procedures which are i) use of bilinear transform for the correction of film distortion and ii) use of lens distortion formulas for the correction of lens distortion.

Having corrected the distortions in the image coordinates, the next stage is to derive 3-D coordinates from the corrected 2-D position. The procedure used to compute 3-D coordinates is photogrammetric intersection with collinearity equation modification described in [1, 2]. The result of this procedure is a list of 3-D coordinates. For the motion estimation algorithm used in our work, it requires two such lists of 3-D coordinates (X, Y, Z) that correspond to each other at two different time instants (t_i and t_{i+1}); however, due to data noise, the majority of these 3-D coordinates are not suitable for estimating the motion parameters between two different time instants. Therefore, we use the technique suggested in [2] to select the best 3-D point sets for motion estimation. It involves two constraints, (1) rigidity between different time instants and (2) uniform point distribution across the object on the image, to accomplish such kind of selection.

At this stage, we have 3-D point sets that correspond to each other at two different time instants, i.e. p_i and p'_i ; $i = 1, 2, \dots, N$ (p_i and p'_i are 3×1 column matrices). Based on these 3-D point correspondences, the motion equation is: $p'_i = R p_i + T + N_i$, where R is a 3×3 rotation matrix, T is a translation vector (3×1 column matrix) and N_i is a noise vector. To compute the motion parameters, we employ the method presented in [2] which consists of two steps:

1. Find rotation matrix R .
2. Then, calculate the translation T by $T = p' - R p$ and the centroid translation T_c as $p' - p$.

Since the translation T is the result of the motion parameters formulation and may not represent an actual physical translation, the centroid translation T_c provides a more realistic translation representation of the rigid object.

2.3 Experimental Results

We now report the results of our motion estimation method (using 3-D correspondences) on eight sets of stereo image pairs, each at two consecutive time instants (t_i and t_{i+1}). These eight sets of images are *time7-8*, *time8-9*, *time9-10*, *time10-11*, *time11-12*, *time12-13*, *time13-14* and *time14-15*.

The matching algorithm is applied to all these image sets to obtain the matched point pairs. Figure 1 serves as an example and depicts the matched results of the image set *time10-11*.

The procedures of the other stages discussed, which include image distortions correction, 3-D coordinates calculation and selection and motion parameters computation, are applied to the matched points of the eight sets of stereo image pairs (*time7-8*, *time8-9*, *time9-10*, *time10-11*, *time11-12*, *time12-13*, *time13-14* and *time14-15*). The estimated motion parameters of the eight sets of stereo image pairs are tabulated as shown in Table 1. We also show the estimated motion parameters with hand picked point correspondences (their Y coordinates in 3-D are set to zero) of the same eight sets (except *time8-9* and *time9-10*) of stereo image pairs in Table 2 (regarded as ground-truth values).

For our comparison, the error E between two translation vectors (\vec{V} , \vec{V}_{gt}) is defined as $E = \left| \frac{\|\vec{V}_{gt}\| - \|\vec{V}\|}{\|\vec{V}_{gt}\|} \right|$ and the angle γ between them is given by $\gamma = \cos^{-1} \frac{\vec{V}_{gt} \cdot \vec{V}}{\|\vec{V}_{gt}\| \|\vec{V}\|}$. Hence, we have the following summary [2]:

1. For the rotation axis ($\hat{n}_1, \hat{n}_2, \hat{n}_3$), the angle γ between the two axes (the estimated and the ground truth) has an average value $\bar{\gamma} = 4.74^\circ$ with standard deviation $\sigma_\gamma = 2.32^\circ$.
2. For the rotation angle α , the average absolute error $|\bar{E}_\alpha|$ is 0.94° with standard deviation $\sigma_\alpha = 0.60^\circ$.
3. For the translation of centroid T_c , the average error \bar{E}_{T_c} is 6.44% with standard deviation $\sigma_{T_c} = 4.08\%$. For the angle γ between the two centroid translation vectors (the estimated and the ground truth), the average value $\bar{\gamma}$ is 8.97° with standard deviation $\sigma_\gamma = 6.97^\circ$.
4. For the translation T , the average error \bar{E}_T is 13.90% with standard deviation $\sigma_T = 8.52\%$. For the angle γ between the two translation vectors (the estimated and the ground truth), the average value $\bar{\gamma}$ is 3.57° with standard deviation $\sigma_\gamma = 1.61^\circ$.

3 Motion Estimation Using Line Correspondences over a Monocular Image Sequence

There are three major stages in motion estimation using line correspondences: straight edge extraction, straight line matching and motion estimation using line correspondences. We shall briefly describe them below. The details can be found in [4].

3.1 Straight Line Extraction

The steps of straight edge extraction are as follows: First, edge support focusing is performed over an intensity image to get an edge support image, which contains the pixels around the significant edges of the original image. Then, the edge support image is segmented into line support regions by analyzing the similarity of gradient orientation of the intensity image. Potentially, one straight edge can be obtained from each line support region. Finally, a line representing the straight edge is computed from the pixels of each line support region.

3.2 Matching Straight Lines

We decompose multi-frame matching into several two-frame matching in our match process. In general, relaxation is often used in the process of matching. However, since straight lines have many significant attributes, such as position and orientation, powerful matching functions [5] can be computed. The matching function is defined as a weighted sum of the differences of the attributes of image lines between the two frames.

A "kernel" method is used in our matching process. With this method, one first search for correspondences between subsets of the lines in image A and in image B. The line correspondences obtained in this step is called the match base or kernel. After kernel matching, a further matching for the remaining lines is performed. In this step, the matching function is estimated based on the lines already matched. Since the matching function is computed only with respect to the already-matched lines, the cost of computation is reduced.

3.3 Algorithm of Motion Estimation

We do motion estimation using line correspondences described in [6]. For the algorithm, we assume that an object is moving in front of a stationary camera in the 3-D world. The images used for motion estimation are taken by the camera at consecutive time instances. The coordinate system of the 3-D world is chosen to be fixed on the camera with the origin coinciding with the focal point of the camera, the Z-axis coinciding with the optical axis and pointing to the front of the camera. The focal length of the camera is assumed to be one unit for simplicity. The image plane is then located at $Z = 1$ with its coordinate axes x, y parallel to the axes X and Y of the 3-D world coordinates respectively. We consider the motion of the 3-D object first from time t_1 to time t_2 , then from time t_2 to time t_3 , $t_1 < t_2 < t_3$. In each time interval, the motion is considered as a 3-D rotation around an axis though the origin followed by a translation. Suppose that p_1, p_2 and p_3 are the positions of an object point in the world coordinates at the instants t_1, t_2 and t_3 ; R_{12}, T_{12} and R_{23}, T_{23} are the rotations and translations from t_1 to t_2 and from t_2 to t_3 respectively; then the motion equations are

$$p_2 = R_{12}p_1 + T_{12}$$

$$p_3 = R_{23}p_2 + T_{23}$$

in which R_{12} and R_{23} are 3×3 right-handed orthonormal matrices and T_{12} and T_{23} are 3-D vectors.

3.4 Implementation and Experiment Results

We only consider the motion of the truck in time intervals of *time13-14* and *time14-15* using the left image sequence.

1. Edge Extraction: In addition to edge support focusing, line support region segmentation and physical edge computation, image geometrical distortion correction is performed in this process. First, lens distortion and film distortion are corrected for the coordinates of the points in line support regions. Then, physical edges are determined from the corrected points of each line support region.
2. Matching: Besides matching function and kernel method, a similarity based technique for background removing is used. Figure 2 shows the result of this stage. A set of 36

line correspondences are obtained over 3 frames.

3. Camera Alignment: For the problem of motion estimation, it is assumed that the camera is stationary. The violation of the assumption in real experiments will lead to erroneous results. Thus, the camera must be aligned for different time instants. To do this, we rotate the image line representations by the matrix representing the camera orientations at each time instant.
4. With the initial guess of rotation axes being $(0, 1, 0)$ and rotation angles being 1° , the computational results are listed in Table 3.
5. Table 4 gives the ground-truth for the motion of the truck also measured in the world coordinate system.

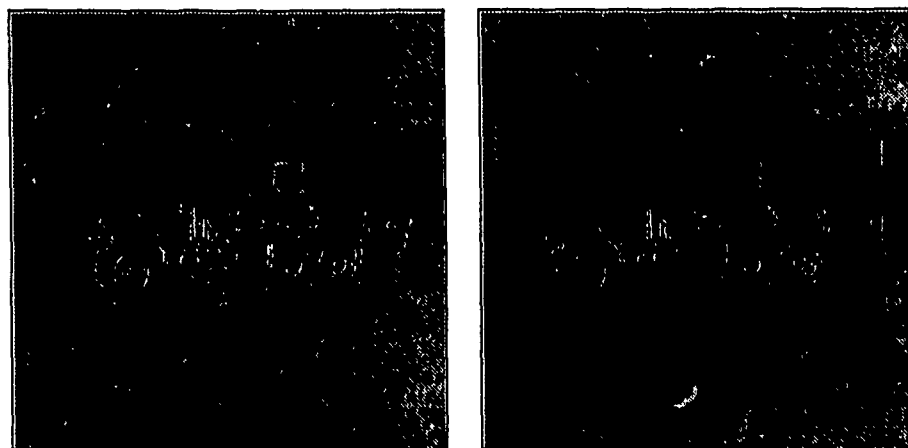
4 Concluding Remarks

Motion estimation algorithms based on stereo image sequences appear to work as well as theoretical analysis indicates. The fundamental limitation appears to be the spatial quantization (sampling) of images. For the typical outdoor scenario we worked with, where the range to baseline ratio is around $10 : 1$ ($35m$ to $3m$), rotation can be estimated with an error of around $10 - 20\%$. The 3-D object location at each time instant, however, can be estimated to only within a meter or so, which makes the translation estimation quite inaccurate.

With the ratio of range to focal length being $900 : 1$, the monocular motion estimation using line correspondences can obtain the rotation with comparable errors. However, the translation estimation is grossly in errors. Monocular point methods are even worse.

References

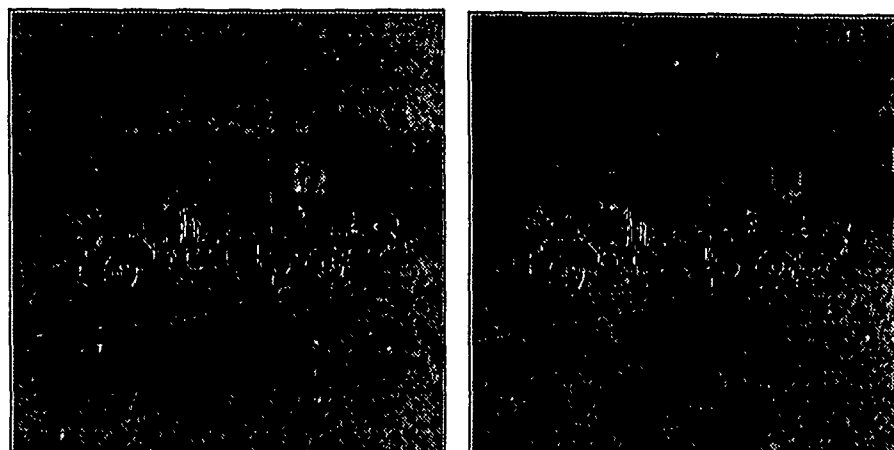
- [1] E. M. Mikhail and F. C. Paderes, "Photogrammetric series of moving vehicle," Scientific Services Program DAAL03-86-D-0001 (0683), CAI-RI, U. S. Army ETL, Fort Belvoir, Virginia 22060-5546, Nov. 1988.
- [2] M. K. Leung and T. S. Huang, "Estimating 3-D vehicle motion in an outdoor scene using stereo image sequences," Tech. Rep. ISP-1010, Coordinated Science Laboratory, University of Illinois at Urbana-champaign, Urbana, IL 61801, Apr. 1990.
- [3] M. K. Leung, A. N. Choudhary, J. H. Patel, and T. S. Huang, "Point matching in a time sequence of stereo image pairs and its parallel implementation on a multiprocessor," in *IEEE Workshop on Visual Motion*, (Irvine, CA), Mar. 1989.
- [4] Y. Liu and T. S. Huang, "Estimating 3-D vehicle motion in an outdoor scene using line correspondences," Tech. Rep. ISP-1005, Coordinated Science Laboratory, University of Illinois at Urbana-champaign, Urbana, IL 61801, Apr. 1990.
- [5] J. H. McIntosh and K. M. Mutch, "Matching straight lines," *Comput. Graphics Image Processing*, vol. 43, pp. 386-408, July 1988.
- [6] Y. Liu and T. S. Huang, "Estimation of rigid body motion using straight line correspondences," *Comput. Graphics Image Processing*, vol. 43, July 1988.



Left

Right

(a) Time instant 10



Left

Right

(b) Time instant 11

Figure 1. The matched points of *time*10-11.

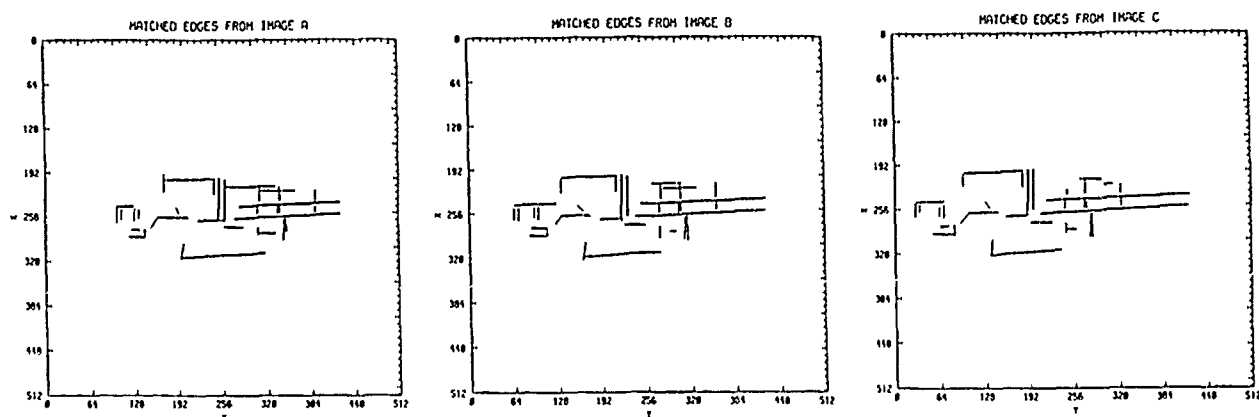


Figure 2. Line Correspondences.

Table 1. Estimated motion parameters of the eight sets of stereo image pairs.

File	Rotation Axis $\hat{\eta}$			Angle α	Translation of Centroid T_c			Translation T		
	$\hat{\eta}_1$	$\hat{\eta}_2$	$\hat{\eta}_3$		X (mm)	Y (mm)	Z (mm)	X (mm)	Y (mm)	Z (mm)
time7-8	-0.1361	0.9906	0.0111	6.17	-931.42	2.20	537.36	-3838.67	-436.00	4004.03
time8-9	-0.0709	0.9949	0.0721	3.74	-989.33	-10.00	478.26	-2773.00	-293.17	2632.98
time9-10	-0.0024	0.9992	-0.0462	5.08	-791.31	-1.33	857.38	-3328.95	102.23	3584.18
time10-11	0.0219	0.9993	-0.0315	8.00	-738.31	-9.09	585.13	-4782.79	216.75	4931.64
time11-12	0.0925	0.9949	-0.0401	4.82	-807.26	-11.76	414.51	-3271.42	321.40	2992.77
time12-13	0.0973	0.9931	-0.0655	6.13	-587.12	3.93	900.55	-3838.46	526.85	3996.81
time13-14	-0.0764	0.9970	-0.0091	5.72	-523.02	-18.46	830.93	-3741.30	-239.98	3607.94
time14-15	0.0137	0.9997	-0.0213	5.45	-544.22	-15.87	778.34	-3587.09	81.81	3406.88

Table 2. Ground-truth for motion estimation.

File	Rotation Axis $\hat{\eta}$			Angle α	Translation of Centroid T_c			Translation T		
	$\hat{\eta}_1$	$\hat{\eta}_2$	$\hat{\eta}_3$		X (mm)	Y (mm)	Z (mm)	X (mm)	Y (mm)	Z (mm)
Y = 0.00										
time7-8	0.00	1.00	0.00	5.23	-945.41	0.00	286.88	-3475.65	0.00	3424.76
time8-9	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
time9-10	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
time10-11	0.00	1.00	0.00	5.91	-743.04	0.00	642.98	-3787.28	0.00	3798.66
time11-12	0.00	1.00	0.00	4.53	-642.27	0.00	757.66	-3089.39	0.00	3056.09
time12-13	0.00	1.00	0.00	5.48	-590.79	0.00	745.81	-3602.12	0.00	3491.67
time13-14	0.00	1.00	0.00	4.50	-562.40	0.00	792.53	-3082.15	0.00	3016.54
time14-15	0.00	1.00	0.00	5.87	-459.67	0.00	864.27	-3796.00	0.00	3751.46

Table 3. Result of motion estimation using line correspondences.

File	Rotation Axis $\hat{\eta}$			Rotation Angle α	Translation		
	$\hat{\eta}_1$	$\hat{\eta}_2$	$\hat{\eta}_3$		X	Y	Z
time13-14	-0.0407	0.9991	-0.0048	4.08	0.8839	0.0174	-0.4674
time14-15	-0.0504	0.9987	0.0055	4.76	1.0400	0.0237	-0.2639

Table 4 Ground-truth motion

File	Rotation Axis $\hat{\eta}$			Rotation Angle α	Translation		
	$\hat{\eta}_1$	$\hat{\eta}_2$	$\hat{\eta}_3$		X	Y	Z
time13-14	0.0000	1.0000	0.0000	4.5	0.9999	0.0000	0.0124
time14-15	0.0000	1.0000	0.0000	5.87	1.3864	0.0000	-0.1067

Deriving Line and Surface Orientation by Statistical Methods

Robert T. Collins and Richard S. Weiss

Computer and Information Science Department
University of Massachusetts at Amherst
Amherst, MA. 01003 *

This paper demonstrates how confidence regions for line and surface orientation can be obtained directly from stereo line correspondences or vanishing point analysis. Since orientations are represented as unit vectors, statistical techniques for estimating the axes and uncertainty of point distributions on the unit sphere are explored. Bingham's distribution is introduced to describe both equatorial and bipolar distributions of unit vectors. Statistical parameter estimation based on Bingham's distribution is used to solve for the polar axis of a great circle of points and to represent the statistical uncertainty in the resulting orientation estimate. In addition, the problem of estimating the orientation and uncertainty of the cross product of two uncertain unit vectors is considered, and an approximate solution is given in terms of the "intersection" of two equatorial Bingham distributions.

1 Introduction

Many objects in man-made environments have planar surfaces. For automatic construction of 3D models it is necessary to recover these surfaces together with an estimate of their uncertainty. Line and plane orientations can be computed whenever the images of parallel 3D lines are observed. Particular examples are vanishing point analysis on monocular images of groups of parallel lines, and the analysis of multiple images of a single 3D line undergoing pure translation with respect to the camera, either as part of translational motion sequence, or from a stereo image pair where the camera axes are aligned.

The representation of orientations as unit vectors leads naturally to an examination of distributions on the surface of the unit sphere, and to statistical inferencing techniques over such distributions. One useful tool for deriving orientation information is the estimation of a unit vector perpendicular to a set of derived unit vectors. For instance, the normal vector to a planar surface is perpendicular to the orientations of all the lines in the plane. For two input vectors, this inference is a cross product

with an uncertainty that reflects the uncertainty in the original vectors. For more than two vectors, the problem can be viewed as finding the polar axis of a great circle defined by the heads of several uncertain unit vectors. Perpendicularity constraints between orientation vectors also lead to an efficient Hough transform technique on the sphere for clustering vectors that lie in a great circle.

Although this paper focuses on the extraction and statistical description of line and plane orientations from stereo line pair correspondences, the methods are applicable in other situations as well. An application of this work to vanishing point analysis will appear in [7], and is briefly outlined in Section 1.3.

1.1 Line Orientations from Stereo

The orientation of 3D lines can be computed directly from stereo line correspondences without first computing point depths, in contrast to methods that compute depth in order to obtain the orientation of line segments. Consider a 3D line segment with unit orientation vector U , projecting onto the image plane of a single camera. The focal point of the camera together with the 3D line defines a plane called the *projection plane* of the line. The image projection of the 3D line lies on the intersection of the projection plane and the image plane, thus the projection plane can be computed given a line segment in the image and the focal point. Since a 3D line lies in its projection plane, the plane normal ϕ is perpendicular to the orientation U . If the same 3D line is imaged from a second camera, oriented the same as the first, but translated by a vector T , a second projection plane that is still perpendicular to U will be measured. The 3D line orientation can thus be recovered as a unit vector parallel to the cross product of the two projection plane normals, except when the line image lies along an epipolar line for the two images.

Looking at this another way, translating the coordinate system by T is equivalent to translating lines by $-T$. Pure translation does not change line orientations, so the new line remains parallel to the original. Under perspective projection, parallel 3D lines of orientation U

*This work was supported in part by DARPA and RADCL under contract number F30602-87-C-0140, by DARPA and U.S. Army ETL under contract number DACA76-89-C-0017.

project to converging image lines which intersect at the vanishing point associated with U . Therefore the image of a line in one camera coordinate system intersects the image of the same line in a translated coordinate system at a vanishing point, from which the line direction can be derived.

Unfortunately, the input line data is always imperfect due to noise in its imaging and extraction, and the effects of noise on the computed 3D orientation must be taken into account. When working with real data, any computed quantity should be treated as an estimate only, with an associated measure of uncertainty. In this instance, a method is needed for computing the distribution of the orientation of the cross product of two random vectors. This problem is addressed in Section 2.2.

1.2 Recovering Planar Surfaces

Having first computed 3D line directions, it is possible to discover coplanar lines and thereby recover the orientation and distance of the planar surfaces that contain them. This is done in two stages. First the lines are broken into groups consistent with a family of parallel planes, then distances are finally computed to partition the lines into sets consistent with individual plane equations.

The normal vector to a planar surface is perpendicular to the orientations of all lines on that surface. Conversely, given a line of orientation U lying on a planar surface, the set of possible surface normals is the set of unit vectors perpendicular to U . On the unit sphere, the heads of this set of possible normals trace out a great circle with polar axis U . This geometric constraint leads to an efficient Hough transform technique for finding possible surface normals on the sphere, a transform first employed by Barnard in the context of locating potential vanishing points [3]. Each line orientation is mapped onto a 2D histogram representing the surface of the unit sphere, partitioned by azimuth and elevation. Each orientation casts a vote in all buckets along the great circle representing vectors perpendicular to it. Potential surface normals are detected as peaks in the histogram, corresponding to areas where several great circles intersect.

Once potential plane orientations have been identified, the unit normal can be estimated with more accuracy. Although the true line orientations lie on a great circle around the true plane normal, this relationship will not be exact due to errors in the derived line orientations. Instead, line orientations lie scattered in an equatorial belt. In the Section 2.1, an estimate of the polar axis of this belt is developed, providing a vector estimate plus an uncertainty region for the 3D surface orientation.

Finally, for lines within a family of parallel planes, a 1D histogram of plane distances is formed by computing for each line a hypothesized distance. The distance is computed as $d_i = \hat{n} \cdot p_i$, where p_i is a 3D point on line i and \hat{n} is the estimated plane normal. Peaks in this 1D

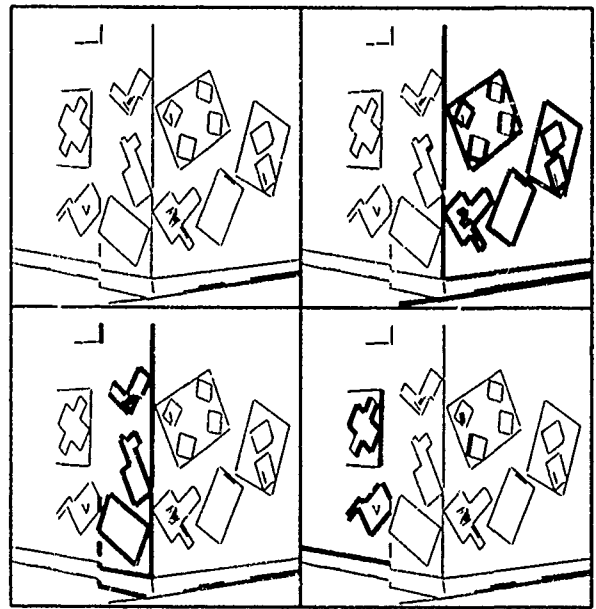


Figure 1: Matched input lines from the left image of a stereo pair are shown upper left; also shown are 3 sets of lines consistent with hypothesized surface planes.

histogram represent sets of lines consistent with a single plane equation.

Figure 1 shows an example of the partition created for a stereo hallway image. The algorithm forms hypotheses of all three visible wall planes, and correctly identifies that one plane orientation is shared by two parallel planes at different depths.

The above method for hypothesizing planes from stereo line correspondences has some unique features. The method typically employed to solve the same problem is to compute 3D line segments or points, then cluster them into planar patches. We believe our approach has the following advantages:

- The computation of plane distance is decoupled from computing plane orientation, thus the Hough transform for detecting surface normals is 2-dimensional instead of 3, and the entire process is roughly $O(N)$ in the number of lines [6].
- Parallel planes are immediately identified, and their shared orientation is computed from all of the lines on them.
- When the depth to a point on a line is finally computed, the line is assumed to lie in a plane with a given orientation that has been estimated from several lines. The depth computed under this constraint is presumably more accurate.

1.3 Vanishing Point Analysis

Several of the techniques described above can also be applied to the analysis of vanishing points. When parallel 3D lines are projected onto an image plane, their images

converge to a common point of intersection called the *vanishing point*. A ray constructed from the camera focal point towards the vanishing point has the same 3D orientation as the original world lines.

It was shown in Section 1.1 that the projection plane normals of parallel lines in 3-space are perpendicular to the 3D line orientation U . This constraint suggests that the 2D Hough transform method of Section 1.2 can be used to find candidate vanishing points [3]. Each projection plane normal votes for a great circle of possible line orientations; potential orientations are then found as peaks in the histogram. Analogous to the situation described for stereo line correspondences, a more accurate estimate of the line orientation perpendicular to a great circle of projection plane normals can be obtained by estimating the polar axis of an equatorial distribution on the sphere [6].

2 Estimates of Orientation

In this section, the statistical problem of representing and estimating orientation vectors and deriving confidence regions for the resulting estimates is examined. Two particular cases are explored: estimating the polar axis of a great circle of unit vectors, and estimating the cross product of two uncertain unit vectors.

Unit orientation vectors really have only two independent values, thus representing the entire vector as a 3D normal variable leads to near-singular covariance matrices. One possibility is to represent unit vector uncertainties as bivariate normal vectors in a local coordinate system on the tangent plane or in spherical coordinates on the sphere surface. However, this approach leads to some messy coordinate system bookkeeping when trying to combine the uncertainties of two or more vectors. We choose instead to represent the uncertainty as a 3D normal variable constrained to conform to the surface of the sphere.

Bingham's distribution is a standard statistical distribution for representing both bipolar and equatorial clusters of points on the sphere [4,9]. It describes a trivariate normal vector with zero mean and arbitrary covariance matrix, conditioned on the length of the vector being unity. Bingham's distribution thus represents the portion of a trivariate normal distribution $N_3(x; 0, \Sigma)$ that intersects the surface of the unit sphere. The covariance matrix Σ can be written as USU^t , where $U = [u_1, u_2, u_3]$ is a 3×3 orthogonal matrix and $S = \text{diag}(\sigma_1^2, \sigma_2^2, \sigma_3^2)$ is a diagonal matrix of variances. Without loss of generality, assume that $\sigma_1^2 \leq \sigma_2^2 \leq \sigma_3^2$. Letting $K = \text{diag}(\kappa_1, \kappa_2, \kappa_3) = -\frac{1}{2}S^{-1}$ and constraining x to have unit length yields the Bingham distribution

$$B(x; K, U) = B(K) \exp\{\text{tr}(KU^t x x^t U)\} \quad (1)$$

where $B(K)$ is the normalizing constant required to make $\int_{x^t x=1} B(x; K, U) = 1$, and $\text{tr}(\cdot)$ denotes the

trace of a matrix.

It is easy to verify that $B(-x; K, U) = B(x; K, U)$, thus the distribution is antipodally symmetric, and it is appropriate to interpret x as an undirected axis, or as an observation confined to the unit hemisphere. Further analysis shows that the "shape" parameters κ_i are determined only up to an additive constant. For uniqueness it is customary to subtract out the largest κ_i , in this case κ_3 , leaving $k_i = \kappa_i - \kappa_3$ with $k_1 \leq k_2 \leq k_3 = 0$. Figure 2 displays the Bingham distribution for varying values of k_1 and k_2 .

Given a set of n unit vectors $\phi_i = (x_i, y_i, z_i)$ assumed to be distributed according to Bingham's distribution, a sufficient statistic for the orientation and shape parameters U and K is the sample second moment or *scatter matrix*

$$M = \frac{1}{n} \begin{bmatrix} \sum x_i^2 & \sum x_i y_i & \sum x_i z_i \\ \sum x_i y_i & \sum y_i^2 & \sum y_i z_i \\ \sum x_i z_i & \sum y_i z_i & \sum z_i^2 \end{bmatrix}. \quad (2)$$

Since the scatter matrix is a symmetric real matrix, it can be decomposed into $M = \Lambda \Lambda \Lambda^t$, where $\Lambda = [a_1, a_2, a_3]$ is an orthogonal matrix of eigenvectors, and $\Lambda = \text{diag}(\lambda_1, \lambda_2, \lambda_3)$ is a diagonal matrix of corresponding eigenvalues with $\lambda_1 \leq \lambda_2 \leq \lambda_3$ summing up to 1. It can be shown [4,9] that the maximum likelihood estimate of the Bingham orientation matrix U is the matrix of eigenvectors Λ . Maximum likelihood estimates of the shape parameters k_1 and k_2 are nontrivial functions of Λ [8].

Recall from Section 1.2 that a set of line orientation vectors perpendicular to the surface normal formed an equatorial belt around the surface normal direction. We will assume the distribution of line orientation vectors can be described as an equatorial Bingham (Figure 2d-e). Finding the planar surface normal reduces then to estimating the polar axis of the distribution. If the convention $\lambda_1 \leq \lambda_2 \leq \lambda_3$ is adhered to, the polar axis of an equatorial Bingham distribution will be u_1 , the eigenvector associated with the smallest eigenvalue λ_1 of the second moment matrix.

After finding the maximum likelihood estimate $\hat{u} = \hat{u}_1 = a_1$ of the desired 3D orientation vector, the construction of a confidence region can begin. Bingham suggests [4] that for large sample sizes, an approximate $1 - \alpha$ confidence region for u_i is an ellipse centered at \hat{u}_i , with axes of lengths $\{\chi_{2,\alpha}^2 / (2n(k_i - k_j)(\lambda_i - \lambda_j))\}^{1/2}$ directed along great circles towards \hat{u}_j , $j \neq i$, where $\chi_{2,\alpha}^2$ is the upper α critical point of the χ^2 distribution with 2 degrees of freedom. For small errors this is approxi-

the set of vectors

$$\left\{ v : v^t R v \leq \frac{\chi_{2,\alpha}^2}{2n} \right\},$$

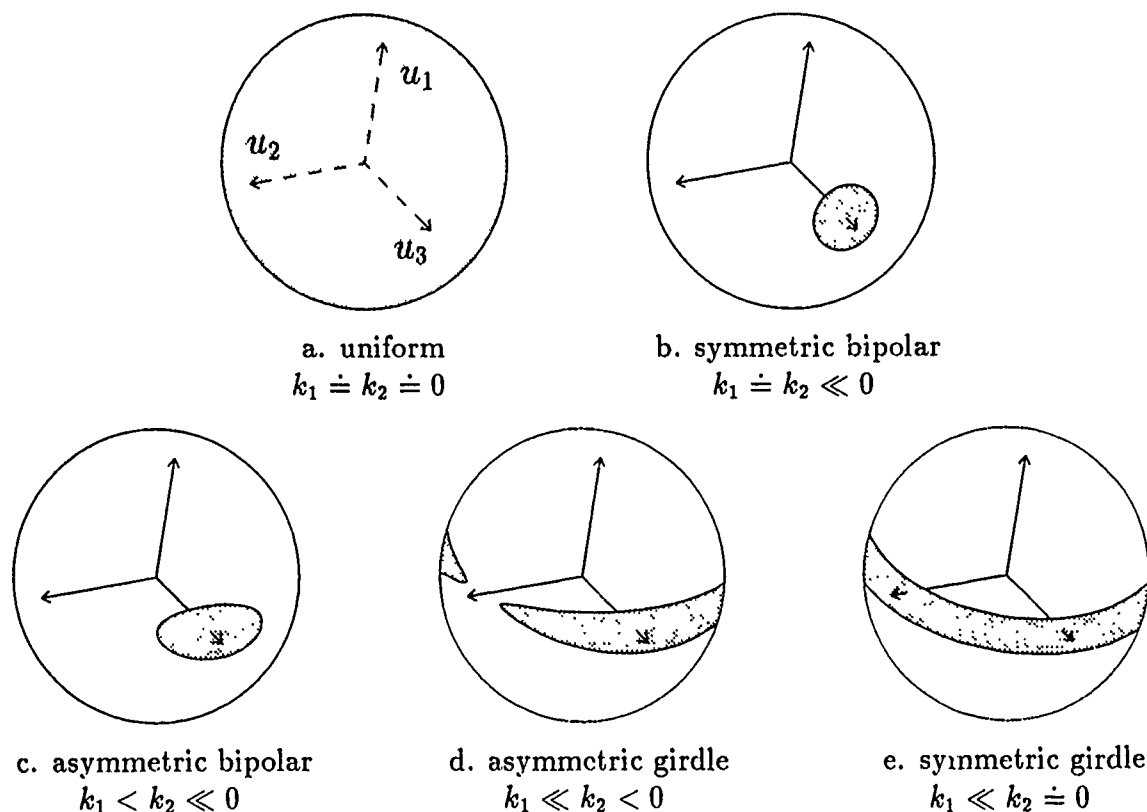


Figure 2: Bingham's Distribution - representative contours for varying shape parameter magnitudes

where R is the matrix

$$[a_2, a_3] \begin{bmatrix} (k_1 - k_2)(\lambda_1 - \lambda_2) & 0 \\ 0 & k_1(\lambda_1 - \lambda_3) \end{bmatrix} [a_2, a_3]^t.$$

2.1 Approximate Confidence Regions

In addition to decomposing the scatter matrix into principle components (eigenvalues and eigenvectors), construction of a Bingham confidence region requires computation of the shape parameters k_1 and k_2 . This process is computationally expensive, however, due to the complexity of the distribution normalization constant [8]. To find a more convenient approximation, first note that a decomposition of the sample scatter into principle components is also found at the heart of the least-squares perpendicular error plane fit [2]. In particular, the eigenvector associated with the smallest eigenvalue of the sample scatter matrix is an estimate of both the pole of the equatorial Bingham distribution, and the normal of the least-squares best-fit plane.

To develop an approximate confidence region, further note that the linear least squares plane fit $z = \alpha x + \beta y$ yields the same plane estimate as the perpendicular error plane fit when the estimated plane is perpendicular to the z axis, since then the distance parallel to the z -axis from a point to the plane is the orthogonal distance [1]. This means that if the original coordinate system is rotated such that the normal of the orthogonal plane fit coincides with the z -axis, a linear least squares plane fit

will also yield the z -axis as the rotated plane normal vector. More importantly, the confidence region associated with this linear least-squares plane fit is an approximate rotated confidence region.

Without detailing the intermediate steps in building a confidence region for a linear least squares solution (see [5], for example), the following approximation to the Bingham confidence region is derived

$$\left\{ v : v^t P v \leq \frac{2 F_{\alpha}(2, n-2)}{n-2} \right\}, \quad (4)$$

where $F(2, n-2)$ is an F variable with 2 and $n-2$ degrees of freedom, and P is the matrix

$$[a_2, a_3] \begin{bmatrix} \lambda_2/\lambda_1 & 0 \\ 0 & \lambda_3/\lambda_1 \end{bmatrix} [a_2, a_3]^t.$$

This region most closely approximates equation (3) for rotationally symmetric, low variance equatorial distributions, as in Figure 2e. Furthermore, approximating Bingham polar axis estimation as a least squares plane fit allows us to include information about non-equal variances among the input vectors simply by substituting the formula for a weighted least squares fit.

2.2 Cross Product Confidence

Recall that the 3D orientation of parallel lines in the world is computed from the cross product of the projection plane normals of their images. A cross product is a great circle pole estimate for two vectors. Unfortunately,

the formula for estimating polar axis confidence regions is not valid for less than three vectors. In this section a formula to compute an approximate confidence region for the cross product is derived, based on intersecting equatorial Bingham distributions.

A geometrical intuition for computing an uncertain cross product is to intersect belts on the sphere, each belt representing all the points perpendicular to the uncertainty region at its pole [10]. In the current formalism, if two equatorial Bingham distributions exist, with distributions $P(\mathbf{x} \sim N(0, \Sigma_1) \mid \mathbf{x}^t \mathbf{x} = 1)$ and $P(\mathbf{x} \sim N(0, \Sigma_2) \mid \mathbf{x}^t \mathbf{x} = 1)$, the probability that a vector lies in the intersection of the two is their joint probability, which assuming independence is the product of the individual probabilities. It can be shown that this joint probability has the Bingham distribution $P(\mathbf{x} \sim N(0, (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1}) \mid \mathbf{x}^t \mathbf{x} = 1)$.

The only remaining task is to construct a "dual" equatorial Bingham distribution from a bipolar one, where dual means that the longitudinal variance across the equator of the equatorial distribution is the same as the longitudinal variance at the pole of the bipolar distribution (see Figure 3). For general asymmetric bipolar distributions this is difficult, but an approximate dual for rotationally symmetric forms exhibiting small variances can be constructed. Under those conditions $k_1 = k_2 \doteq -1/(2\sigma^2)$, where σ^2 is the variance across the pole [4]. If the axis of the pole is \mathbf{u}_3 , the matrix of shape parameters \mathbf{K} becomes $\text{diag}(0, 0, -1/(2\sigma^2))$, and $\Sigma^{-1} = -2\mathbf{U}\mathbf{K}\mathbf{U}^t$. Decomposition into principle components of the sum of Σ_1^{-1} and Σ_2^{-1} yields the orientation and shape of the resulting joint Bingham.

2.3 Numerical Results

The stereo line correspondence example from the last section was taken with a two-camera setup with parallel focal axes and a baseline of 20 inches. The distance from the left camera to the corner was approximately 18 feet. As described in Section 1.1, each line orientation was computed as the cross product of two corresponding projection plane normals. The variance in each input normal was taken to be circular, and inversely proportional to the average length of the matched lines in the image. From these two uncertain projection plane normals, the line orientation was computed as their cross product, and an associated confidence region was computed as described in Section 2.2. Although this generally gave an elliptical confidence region, uncertainty in the line direction was taken to be a circular distribution circumscribing the elliptical one on the sphere, summarizing the region description with a single variance.

Two plane orientations were discovered from the Hough transform array (two planes of the three share the same orientation). To compute plane orientations from each line group found, a weighted least-squares plane fit

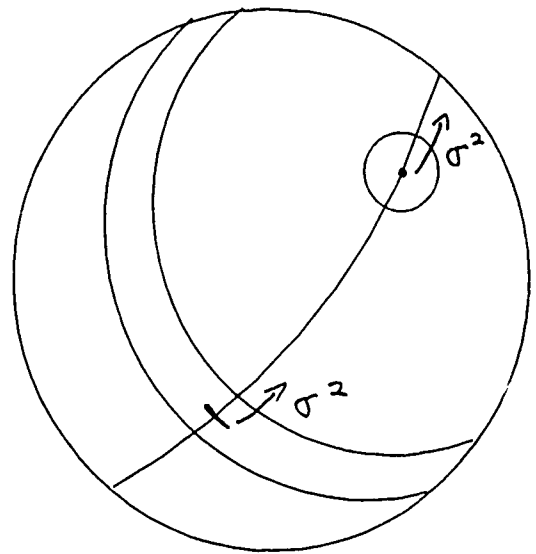


Figure 3: An equatorial Bingham distribution E is the "dual" of a bipolar one P if the variance along a line of longitude across the equator of E is the same as the variance along that line of longitude at the pole of the P .

was applied to the line directions in the group, using a weight inversely proportional to the line direction variance. Approximate confidence regions were computed using equation 4.

The mean orientation vector for the right wall was computed to be $(.665, -.061, -.744)$, as depicted in Figure 4a. A 95% confidence region around the right wall orientation is approximately an ellipse, with half-lengths on the sphere of 2.48 and 2.76 degrees. The shared orientation of the two left walls was estimated to be $(-.739, -.021, -.673)$, with a 95% confidence region of half-lengths 2.53 and 2.0 degrees. The relative orientation between the two estimated orientations is 89.4 degrees, and the actual walls are in fact perpendicular within the usual limits of construction accuracy.

In partitioning the lines into individual plane equations, the two parallel left walls were separated. The perpendicular distance from the camera to the extended plane of the far left wall was computed as 155.16 ± 2.45 inches, and that of the closer left wall as 146.06 ± 1.82 inches (see Figure 4b). This gives a nominal difference of 9.1 inches, whereas the faces of these two walls are in fact 8.5 inches apart, or a 7% error in depth for the estimated value. The wall on the right hand side of the image was estimated to be $162.2 \pm .98$ inches away.

References

- [1] L. Ammann and J. Van Ness, "A Routine for Converting Regression Algorithms into Corresponding Orthogonal Regression Algorithms," *ACM Trans.*

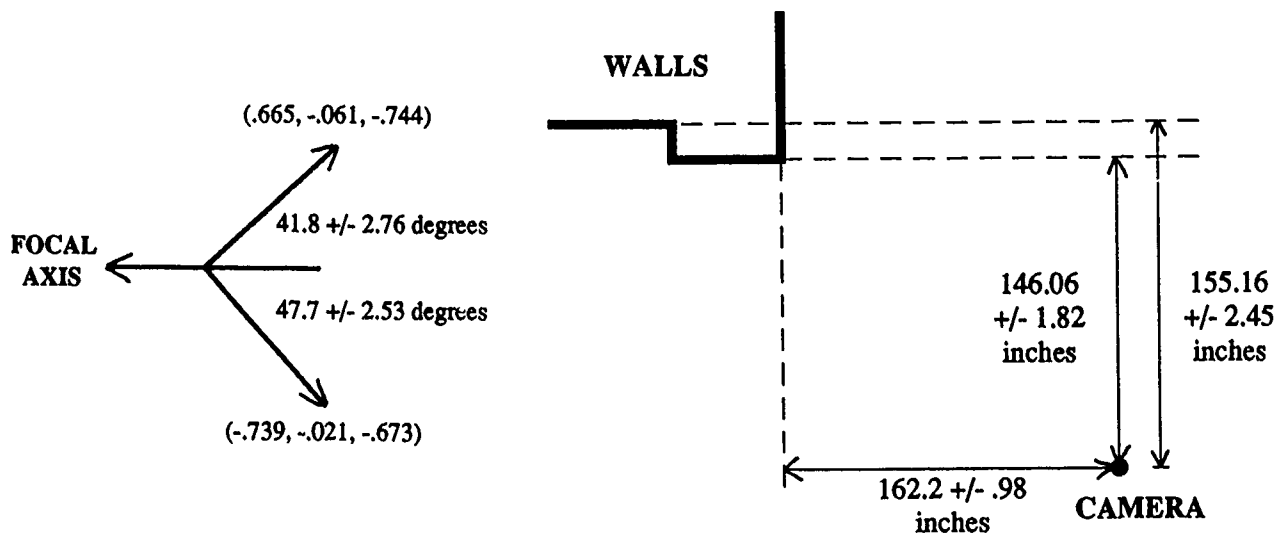


Figure 4: Numerical results for planar surface hypotheses. Left depicts the orientation and uncertainty in surface normals. Right shows perpendicular distance estimates to each plane.

actions on Mathematical Software, vol. 14, no. 1, March 1988, pp. 76-87.

ishing Points," to appear in *Pattern Analysis and Machine Intelligence*.

- [2] D.H. Ballard and C.M. Brown, *Computer Vision*, Prentice-Hall, Inc, New Jersey, 1982, pp. 486-487.
- [3] S.T. Barnard, "Interpreting Perspective Images," *AI Journal*, Vol. 21, No. 4, November 1983, pp. 435-462.
- [4] C. Bingham, "An Antipodally Symmetric Distribution on the Sphere," *The Annals of Statistics*, Vol. 2, No. 6, 1974, pp. 1201-1225.
- [5] G.E.P. Box and N.R. Draper, *Empirical Model-Building and Response Surfaces*, John Wiley and Sons, New York, 1987.
- [6] R.T. Collins and R.S. Weiss, "An Efficient and Accurate Method for Computing Vanishing Points," Workshop on Image Understanding and Machine Vision, 1989 Technical Digest Series, Vol. 14, (Optical Society of America, Washington, D.C.) pp. 92-94.
- [7] R.T. Collins and R.S. Weiss, "Deriving Line and Surface Orientation by Statistical Methods on the Unit Sphere," COINS Technical Report, in preparation, University of Massachusetts, Amherst, MA., 1990.
- [8] J.T. Kent, "Asymptotic Expansions for the Bingham Distribution," *Applied Statistics*, Vol. 36, No. 2, 1987, pp. 139-144.
- [9] K.V. Mardia, *Statistics of Directional Data*, Academic Press, New York, 1972.
- [10] R.S. Weiss, H. Nakatani, and E.M. Riseman, "An Error Analysis for Surface Orientation from Van-

Computational and Biological Models of Stereo Vision

Stephen T. Barnard and Martin A. Fischler

Artificial Intelligence Center
SRI International
333 Ravenswood Ave., Menlo Park, CA 94025

Abstract

As part of the SRI research effort involving stereo compilation, we both contribute to and monitor other work relevant to the state of the art in this field. This paper is a survey, intended for a general technical audience, of both computational and biological approaches to stereo vision.¹

1 Introduction

Two eyes or cameras looking at the same objects from different perspectives provide the means to determine three-dimensional shape and position. Scientific investigation of this effect (called variously stereo vision, stereopsis, or single vision) has a rich history in psychology, biology, and, more recently, in the computational modeling of perception. The human visual ability to perceive depth is both commonplace and puzzling: one perceives three-dimensional relationships effortlessly, but the means by which one does so are largely unknown and hidden from introspection. Stereo vision, however, is one way to perceive depth that is relatively well understood from a computational standpoint. Stereo is an important method for machine perception because it leads to relatively direct measurements and, unlike monocular techniques, does not infer depth from weak and unverifiable photometric and statistical assumptions, nor does it require specific detailed models of objects. Once two stereo images are brought into point-to-point correspondence, recovering depth by triangulation is straightforward.

We begin with a discussion of the geometrical basis for stereo vision. We then focus on three distinct computational models, selected to represent both differences and common themes in approach. Next, we discuss some aspects of biological stereo vision and practical applications of computational stereo. Finally, we conclude by identifying the important open questions.

¹Some of the work described in this article was supported under DARPA contracts MDA903-86-C-0084 and DACA76-85-C-0004. The article will appear in the Wiley Encyclopedia of Artificial Intelligence (2nd edition) to be published in the latter part of 1991.

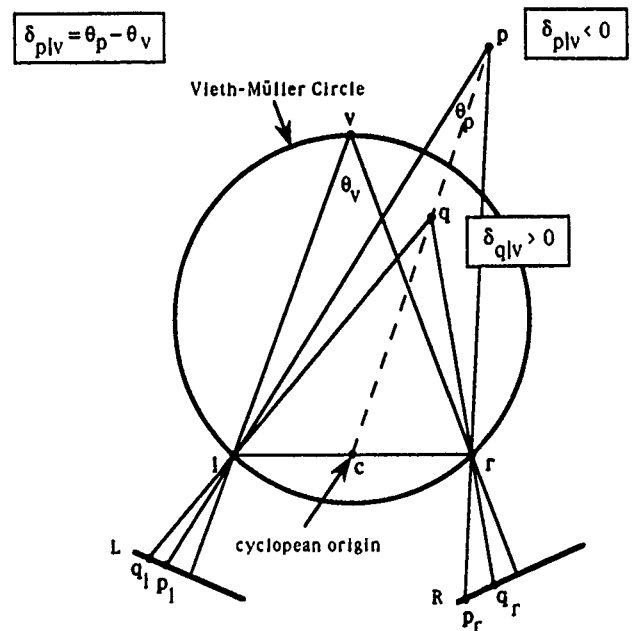


Figure 1: Basic Stereo Geometry.

2 Stereo Geometry

The geometrical principle behind stereo vision, illustrated in Figure 1, is quite simple. Assume that two cameras form images through left and right centers of perspective l and r, onto planes L and R. (In practice these would be imperfect optical lens systems, but for this discussion we assume ideal "pinhole" projections.) Furthermore, assume that the cameras are fixed upon point v, which is to say that the two rays perpendicular to the image planes passing through the centers of perspective (the *principle rays*) intersect at v. Let θ_v be the angle between these principle rays. We say that the *absolute disparity* of v is θ_v . Now consider another point p projected onto image planes L and R as shown, and let the angle between these rays be θ_p . We say that the *relative disparity* of p with respect to v is $\delta_{p|v} = \theta_p - \theta_v$. Relative disparity is the more commonly used definition

The circle through l , r , and v (actually a sphere) is called the *Vieth-Müller circle* (closely related to the horopter discussed below in Section 4.2). Note that relative disparity δ_{plv} is positive for points inside, negative for points outside, and zero for points on the circle. Positive and negative relative disparities are sometimes referred to as *crossed* and *uncrossed*, respectively.

Clearly, disparity is related to distance. Let $c = (l + r)/2$ be the midpoint between the focal points (the *cyclopean origin*), and p and q be two points on a line with c ; if $|p - c| > |q - c|$ (i.e., p is farther from the observer than q) then $\delta_p < \delta_q$. The two projections of a single point in the scene are called *conjugate points*, and they determine the location of the point in the scene uniquely.

This property of conjugate points establishes the importance of the *correspondence problem*: how can one determine which pairs of points in the two images correspond to actual points on surfaces in the scene? The scene is presumably made of light-reflecting surfaces, and individual markings on these surfaces will, in some sense, look about the same in the two images. In realistic imagery the correspondence problem is much harder than it may first appear, however, due to many complicating factors such as occlusion (in which some parts of the scene are seen only by one camera), periodic surface markings, surface areas with no markings at all, distortion of the surface markings in the images due to perspective projection, sensor noise, optical distortion, etc.

Knowledge of the relative orientation of the two cameras makes the correspondence problem much more tractable by reducing the dimensionality of the search space. Any point in three-dimensional space, together with the centers of projection of the two camera systems, defines a plane called an *epipolar plane* (Figure 2). The intersection of an epipolar plane with an image plane is called an *epipolar line*. Every point on an epipolar line in one image must correspond to a point on the corresponding epipolar line in the other image. This constraint, often called the *epipolar constraint*, therefore limits the search for the match of any point in the first image to a one-dimensional neighborhood in the second image, as opposed to a two-dimensional neighborhood, with an enormous reduction in computational complexity.

3 Computational Models

Many computer models of stereo vision have been proposed. In some cases these models have been offered as a theory of human stereo vision; in other cases the models have been pragmatically task-oriented and unconstrained by knowledge of biological vision. In this section we examine three distinct approaches illustrated by characteristic examples. We first examine the issues

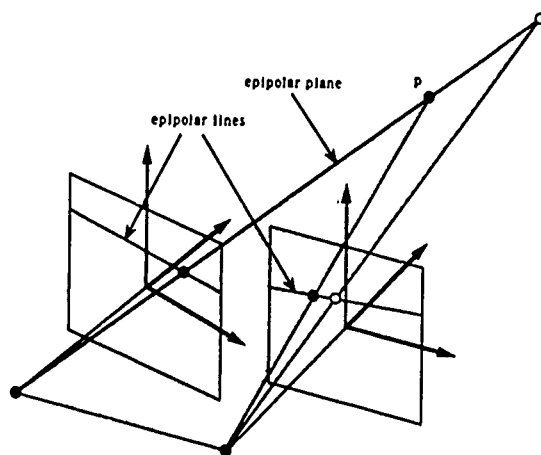


Figure 2: The epipolar constraint. Any other point in the epipolar plane (for example, the point shown as an open circle) must project to image locations on the epipolar lines.

that confront any stereo model to a greater or lesser degree: image acquisition, camera modeling, feature acquisition, image matching (the correspondence problem), depth determination, and interpolation.

image acquisition. Stereoscopic images can be recorded in a large variety of ways. For example, they may be recorded either simultaneously or in a time sequence of any duration; from very similar or from very different perspectives; with accurate, well-calibrated instruments or with a crude hobbyist camera. The most important factor affecting image acquisition is the specific application for which the stereo computation is intended. (See Section 5 for more on applications.)

camera modeling. We have seen that knowledge of disparity allows one to calculate depth, assuming full knowledge of the geometrical arrangement of the cameras. In practice, this means one must know both the *interior orientation* of each camera (the relationship between points in the image and the coordinate system of the camera) and the *exterior orientation* of the two-camera system (the relationship between the local camera coordinate systems and an invariant world coordinate system). Exterior orientation requires knowledge of the locations of the focal points and the orientations of the cameras in the world. A stereo camera model is normally separated into an *absolute* component that specifies the transformations between a camera coordinate system and a world coordinate system, and a *relative* component that specifies the transformations between two camera coordinate systems without reference to a world coordinate system. Knowledge of the relative camera model alone is sufficient to exploit the epipolar

constraint to simplify matching, as described in Section 2. The absolute model is required to translate disparity measurements into absolute measurements of the positions of points in the world.

The derivation of a relative camera model given a large number of matched points is usually accomplished by least-squares parameter estimation. Gennery [10] has developed such a method for finding the relative camera model in terms of azimuth, elevation, pan, tilt, roll, and focal length. From a theoretical standpoint, the minimum number of conjugate points that is necessary to derive a unique relative camera model is 5, but the solution involves three simultaneous nonlinear equations [21]. If as many as eight conjugate points are available, the relative camera model can be obtained directly as the solution of a system of linear equations [18,25]. Fischler and Bolles [9] have provided a number of results with respect to the minimum number of points needed to obtain an absolute camera model, given a single image and a set of correspondences between points in the image and the points' locations in space; they also provide a technique for finding the complete camera model when the given set of correspondences contains a large percentage of errors.

preprocessing Many stereo systems perform some sort of preprocessing of the images before matching, either to set the stage for feature acquisition (see below) or to build image hierarchies (e.g., pyramids) for coarse-to-fine matching algorithms. Preprocessing is typically a linear filtering operation in which the raw image intensities are convolved with one or more digital kernels. The three filters illustrated in Figure 3 are commonly used: the Gaussian, the difference of Gaussians (DOG), and the Laplacian of the Gaussian ($\nabla^2 G$). The Gaussian is lowpass, while the difference of Gaussians and the Laplacian of the Gaussian are bandpass. These circularly-symmetric filters are parameterized by σ , the space constant of the Gaussian. By using a series of filters with different space constants, one can construct a series of images restricted to different spatial frequencies. The lower-frequency images can be subsampled into smaller images to create a pyramid-like resolution hierarchy. Such hierarchies are extremely useful because the range of possible disparity values across the field of view decreases linearly with the sample spacing, which implies that the number of possible matches is relatively smaller at lower resolutions. Coarse matching can be done quickly at low resolution and the result used to initialize the search at the next finer scale.

feature acquisition. Clearly, some parts of the images will be easier to match than others because of the presence of characteristic surface markings. Many computational stereo models select sets of discrete features in the two images and then seek, at least, a sparse set of correspondences between these feature sets. Features can be

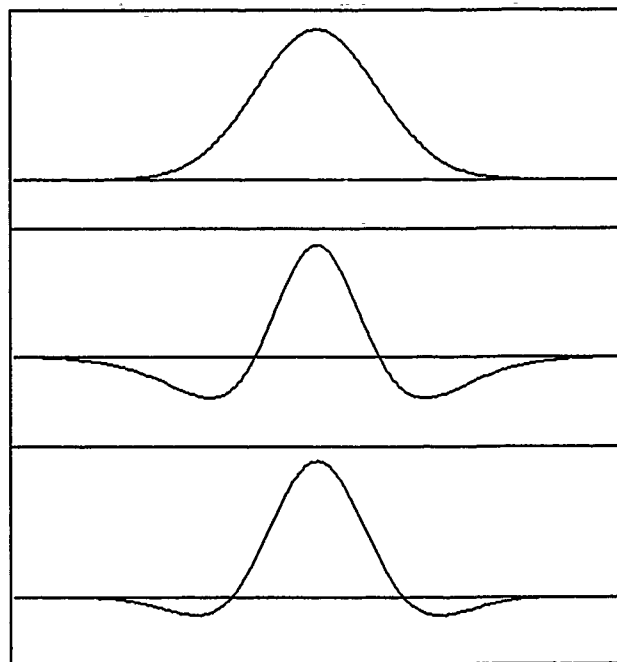


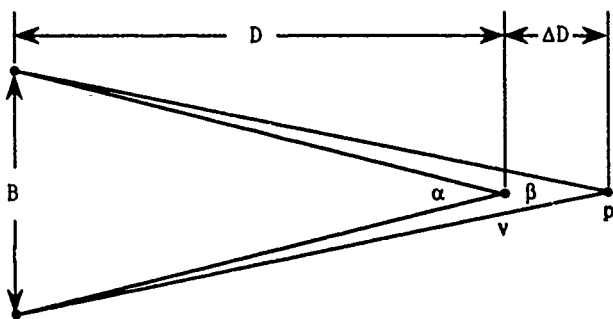
Figure 3: Three common filters for stereo preprocessing. The Gaussian (top), the difference of Gaussians (middle), and the Laplacian of the Gaussian (bottom)

used either as elementary tokens for matching or as "interest operators," indicating the locations of promising correlation windows. Many kinds of features have been used; for example:

- Isolated point-like features can be found by a variety of methods, often modeled along the lines of Moravec's interest operator [20], which selects points where there is high variance in four directions.
- Edge features can be detected by a variety of methods. One commonly used method is to select *zero-crossings* in bandpassed images; that is, those contours where the bandpassed images change sign.
- Higher-level features, such as line junctions, specifically shaped "primitives," and segmented regions are sometimes used.

image matching. The matching step — solving the correspondence problem — is clearly a critical component of any stereo model. There are three distinct approaches:

- **Area matching.** Areas of image-intensity measurements (possibly after preprocessing) are the basic units to be matched. This typically involves using a feature detector to select promising areas,



$\delta_{plv} = \beta - \alpha$ is the disparity of p with respect to v .

Assume that α and β are small: $\alpha \approx B/D$, $\beta \approx B/(D + \Delta D)$

and $D \gg \Delta D$.

Then

$$\delta_{plv} \approx \frac{-B \Delta D}{D^2}$$

Figure 4: Disparity as a function of depth and baseline.

and then using a selected "area patch" in a template search to either maximize a measure of cross-correlation or minimize RMS error.

- **Feature matching:** Discrete features are the basic units to be matched. Once the features have been detected no reference is made to the underlying photometry. In addition to location, features may have other properties, such as contrast sign, contrast magnitude, and directionality.
- **Variational models:** A set of matches is sought by explicitly minimizing a cost function. For example, the cost function may combine a measure of photometric error with a measure of the complexity of a dense disparity map.

In all three approaches hierarchical scale-space techniques, based on the pyramid structures described in preprocessing above, have had success.

depth determination. Once accurate matches have been found, the determination of distance is a relatively straightforward matter of triangulation using the absolute camera model. Nevertheless, this step can present significant difficulties if the matches are somewhat inaccurate or unreliable or if the camera model is uncertain. Figure 4 illustrates how disparity measurements are related to depth and the separation of the cameras. Suppose the cameras are fixated on point v . Assuming small

angles, the disparity of another point p with respect to v is

$$\delta_{plv} \approx \frac{-B \Delta D}{D^2},$$

where B is the distance separating the cameras (the baseline), D is the distance from the cyclopic origin to the fixation point, and ΔD is the difference in the depths of p and v . To a first approximation, the error in stereo distance measurements is inversely proportional to the baseline and directly proportional to the square of the distance to the point being measured. Lengthening the baseline reduces error, but complicates the matching problem by increasing both the disparity in the images and the difference in appearance of the areas or features being matched. Given a constant image size and sampling rate on the focal plane, increasing the focal length will decrease error proportionally, but at the cost of decreasing the field of view.

interpolation. Stereo applications usually demand a dense array of depth estimates. The variational models can produce a dense array directly, but feature matching and area matching can only produce sparse correspondences. (This is obvious for feature-matching. Area matching cannot produce correspondences unless there is significant photometric variation in the areas to be matched.) Consequently, both approaches usually require an interpolation step. The most straightforward way to derive a dense depth array from a sparse one is simply to treat the sparse array as a sampling of a continuous function using a conventional interpolation method. More elaborate and specialized methods have been proposed for interpolation when the depth may be discontinuous. An additional approach to interpolation is to fit prior geometric models to the sparse depth array.

We will now describe three stereo systems in detail. (There are many other models, see the survey articles [1], [5].) These three systems were chosen to illustrate the range of computational approaches to stereo, and also to identify common themes — the use of a scale hierarchy, use of the continuity of depth and disparity to resolve ambiguity, and the use of the epipolar constraint to limit search. These are also well-tested systems that have been used in real applications.

3.1 STEREOSYS

The goal of STEREOSYS [13] is to find as many reliable point-to-point matches as possible in an approximately aligned stereo pair. Matches are determined by maximizing the cross-correlation of image intensities, normalized by mean and variance, between two rectangular areas (typically 11 by 11 pixels). STEREOSYS is an eclectic approach to area matching: several effective techniques have been combined into one system. The strategy is to use different matching algorithms in sequence, beginning with the more general algorithms to get a relatively

few matches with very high confidence, and then using this information to constrain further matching with more specifically tuned algorithms.

The first step is to select a set of well-scattered points in one image that indicate the centers of promising areas for which to find matches in the second image. This is done with an "interest" operator, applied over a small windows, that penalizes windows with little information (i.e., small variance of image intensity) or whose only information is contained in strongly linear features (both conditions that cause problems for correlation matching). The local maxima of this measure identify the set of feature points.

The first matching algorithm, *unconstrained hierarchical matching*, operates over a Gaussian image hierarchy by tracing the locations of the feature points (located in one of the full-resolution images) through the hierarchy of lower-resolution, low-passed images. Starting with the lowest resolution, this algorithm finds the best matches by searching for disparities that maximize a normalized cross-correlation of image intensity. It then uses the result to begin a more constrained search at successively higher resolutions. In this way the system bootstraps itself up to an acceptable set of matches at the highest resolution. A *back-matching* technique is used to confirm each match by repeating the search, starting with the opposite image. The next algorithm, *epipolar constrained hierarchical matching*, uses the highly reliable matches to derive a relative camera model. Feature points that were not matched in the first stage (perhaps because they were not promising enough) are now matched through the image hierarchy using the epipolar constraint. The third algorithm, *anchored matching* uses the matches found so far to establish a continuity constraint for further matching.

3.2 MPG (Marr, Poggio, Grimson)

The MPG model [10,11,12] was motivated by a desire to model the human visual system, at least approximately. It has been modified since it was first proposed — here we shall consider only the most recent description [12]. Briefly, the MPG model matches discrete, oriented edge features from epipolar-corrected images across a resolution hierarchy, resolving ambiguity by enforcing continuity of disparity. The use of the epipolar constraint is justified by the effect of eye movements to align retinal images, the use of the hierarchy is justified by the existence of independent channels tuned to different spatial frequencies in the human visual system, the choice of the feature detector by the existence of similar receptive fields in neural structures, and the use of the continuity constraint by the existence of "cooperative" processes in the brain.

The first step is to preprocess both images by convolving them with a series of bandpass filters. MPG uses

the Laplacian of a Gaussian kernel ($\nabla^2 G$). As described above (preprocessing), the choice of the space constant of G controls the center of the passband; the MPG filters are separated by about one octave. Zerocrossings are selected in the filtered images, and are classified according to contrast sign.

Matching begins at the lowest frequencies, as in STEREOSSYS. Each zerocrossing point in the left bandpass-filtered image is compared to all zerocrossing points in the right image within a fixed distance of the left point's coordinates, corresponding to the maximum range of disparity. (A small amount of vertical disparity is allowed to permit an approximate epipolar camera model.) Those pairs of points that have the same contrast sign are considered as matches. Next, a *figural continuity* constraint is used to cull false matches from the set of possibilities. The basic idea behind figural continuity is that, because zerocrossing contours are most likely to be caused by continuous features in the scene, the correct matches should lead to continuous features in space. Those subsets of matches that produce only very short linear features are discarded. In the event that only one match remains for a point, the disparity of the feature is determined, but if more than one match is found the ambiguity is resolved by selecting the most common disparity in a local neighborhood.

3.3 CYCLOPS

CYCLOPS [2] computes a dense disparity map by posing the correspondence problem as a combinatorial optimization: find the simplest (i.e., flattest) map with the least photometric error. Like STEREOSSYS and MPG, CYCLOPS operates across a sequence of resolutions, in this case a Laplacian pyramid. The Laplacian pyramid is a particularly convenient structure because it is constructed of DOG filters with passbands separated by one octave, quantitatively very close to the $\nabla^2 G$ bandpass filters of MPG. Furthermore, the lower-frequency images have larger sample spacings, and hence are smaller images, leading to faster search at low resolution as in STEREOSSYS. The Laplacian pyramid can be computed very efficiently by exploiting the separability of the Gaussian and by using a small recursive kernel, as described by Burt [4].

At each level of the hierarchy CYCLOPS selects a disparity map by minimizing a discretized form of the *stereo constraint equation*:

$$\mathcal{E} = \int \int \left\{ \left(\mathcal{L}\left(x - \frac{\mathcal{D}}{2}, y\right) - \mathcal{R}\left(x + \frac{\mathcal{D}}{2}, y\right) \right)^2 + \lambda |\nabla \mathcal{D}|^2 \right\} dx dy,$$

where \mathcal{L} and \mathcal{R} are the DOG-filtered image intensities, $\mathcal{D} = \mathcal{D}(x, y)$ is a cyclopean disparity map, and λ is a constant. Note that the epipolar constraint is used because only horizontal disparities are allowed. (The system uses a camera model to perform the epipolar correction if nec-

essary.) \mathcal{E} is a combination of two terms, the photometric error associated with the map and a measure of the first-order variation of the map.

A multigrid simulated annealing algorithm is used to minimize \mathcal{E} . The annealing algorithm uses random numbers to model heat. By slowly cooling the system to zero temperature, an approximation to the optimal disparity map is constructed each level of the pyramid. Simulated annealing is more robust than a deterministic gradient-descent search because it allows "uphill" state transitions, and therefore the possibility of escape from the local optima that one is likely to encounter in nonlinear objective functions such as the stereo constraint equation. Using the theory of Markov random fields, simulated annealing can be interpreted as maximum a *posteriori* estimation. This Bayesian interpretation shows that by minimizing \mathcal{E} one finds the most probable disparity map \mathcal{D} given \mathcal{L} and \mathcal{R} , subject to a prior probability distribution of \mathcal{D} . The *a priori* distribution of \mathcal{D} is given by $\lambda|\nabla\mathcal{D}|^2$, the measure of first-order variation. Disparity maps are considered to be more probable when they are flatter.

Starting with a *base disparity* of zero at the lowest resolution of the pyramid (typically 32x32), an *incremental disparity* map with values in $\{-1, 0, 1\}$ is found that minimizes \mathcal{E} for the *full disparity*, which is the sum of the incremental and the base disparity. Note that the incremental disparity is interpreted as a correction to the base disparity, which is inherited from the previous, coarser level (or which is zero at the coarsest level). To set the base disparity for the next higher resolution, the disparity range is expanded by a factor of 2 and the values of the full disparity are doubled, reflecting the change of spatial scale. Even though the incremental disparity can only assume three values at any level of the pyramid, the full disparity can increase by a factor of 2 across levels.

Because of the Laplacian pyramid structure and the locality of the interactions in the stereo constraint equation, the CYCLOPS algorithm is ideally suited to massively parallel SIMD (single instruction, multiple data) architectures. The current implementation (on a Connection Machine with 4096 processors) can produce dense elevation maps of 1024x1024 aerial images in about eight minutes.

4 Aspects of Biological Stereo Vision

4.1 The Neuroanatomy of Stereo

Much is known about the biological and psychological basis for stereo vision, but there is currently no complete explanation for how the brain exploits binocular information. (See Kandel [16] and Kuffler [17] for more detailed information.) The two retinas of the human visual

system signal their response to visual stimuli through the optic nerves. The optic nerves partially cross in the optic chiasma, carrying information from the left half of the visual field to the right half of the brain, and information from the right half of the visual field to the left half of the brain. The visual field projects to an area on the back surface of the brain called the primary visual cortex (also called striate cortex or area 17). The primary visual cortex is almost certainly the place where the first stereoptic computation is performed.

The visual cortex is a sheet of half a dozen distinct layers of cells. It is morphologically uniform across its entire expanse; groups of cells on one part of the sheet appear essentially the same as groups of cells anywhere else on the sheet. The visual stimulus is *topographically mapped* to the cortex: each part of the cortex responds to stimulation at specific location on the retina, and there are no discontinuities except for the separation of the visual field into left and right halves.

Visual stimulation enters the cortex at one layer (layer IV), and then projects to other layers *locally*, spreading only about 1mm to neighboring cortical cells. The cells in layer IV, (so-called "simple" cells) are strictly monocular; they respond exclusively to one input from one eye. The "complex" cells in the subsequent layers, however, can be stimulated by both eyes to greater or lesser degrees. Globally, the visual cortex is organized into *ocular-dominance columns*. The preference for one eye (a complete preference in layer IV) extends through the width of the cortex (i.e., through a column of cells normal to the surface of the cortex), and eye-preference is also highly correlated between neighboring columns. Ocular dominance columns are thought to play an important role in stereopsis, but the mechanism is not understood. To a large degree this organization is innate, but it does undergo further elaboration and sharpening after birth. Depriving one eye of stimulation after birth causes the formation of deviant ocular dominance columns [14] dominated by the active eye. In addition, there is a psychological effect that is probably related: the development of stereo vision passes through a critical period of postnatal development. If an animal is not exposed to stereoscopic stimulus during this relatively short period, it will never be able fully to develop stereo perception.

4.2 Limits to Stereo Fusion and Depth Perception

About 95% of people with otherwise normal vision have the ability to perform stereopsis. As discussed in Section 3, the accuracy of stereo measurement falls with the square of distance. At 100 ft. human stereo depth perception is accurate to only about 25 ft. The maximum distance at which human stereo vision is possible is about 1500 ft.

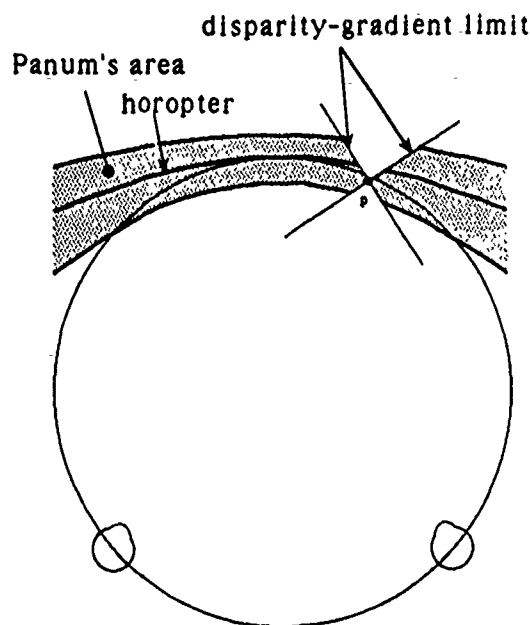


Figure 5: Limits to fusion.

The locality of interaction on the cortex apparently limits the fusion of the two visual fields in human stereopsis. The range over which fusion occurs is called *Panum's area* (Figure 5) [22]. Points inside Panum's area are seen as single points (fusion), while points outside are seen double (diplopia). It is known that while fusion is an important feature of stereo vision, it is not absolutely necessary — even diploptic stimuli can be a cue for depth within limits. The largest disparity that still gives rise to a single, fused image varies from 6 minutes of arc at the center of the visual field to 20 minutes of arc at a peripheral angle of 6 degrees [8]. The thresholds for diploptic stereo perception are about four times those for fusion. The center of Panum's area, shown in the figure as a curve passing through the fixation point, is called the *horopter*. Points on the horopter project to identical positions on the two retinas (i.e., they have zero disparity). Note that the horopter does not follow the Vieth-Müller circle precisely because the geometry and optics of the ocular system don't quite satisfy the ideal model of Figure 1.

Experiments with stabilized images, in which the subjects were prevented from using eye movements to change vergence, have revealed that the limits to fusion are actually much more complex than Panum's area [8]. For example, fusion exhibits a time dependence, called *hysteresis*: as the disparity of fused, stabilized images is increased, they remain fused far beyond Panum's area, until they suddenly "break away" into diplopia. Also, the limits to fusion for stimuli rich in features, such as random dot stereograms (see Section 4.3), is much greater than for simple stimuli such as individual lines.

Another limit on fusion is the *disparity-gradient limit* [3], which states that for fusion to occur the rate of

change of disparity with respect to visual angle (a dimensionless number) must be bounded. Most observers have a limit of about one. While Panum's area is an absolute limit of fusion, the disparity-gradient limit is more subtle. It implies that fusion of one point excludes the possibility of fusion of some nearby points, even if they fall within Panum's area. As shown in Figure 5, if a point *p* is fused then no points in a cone centered on *p* can be fused simultaneously. It has been shown [26] that if an imaged surface has disparity-gradient limit of less than two, then the left and right images are *topologically equivalent*; that is, all points in one image are visible in the other (i.e., the surface is not self-occluding), and furthermore the ordering of points is the same in both images. The disparity-gradient limit has been used to limit search for correspondences in at least one computational model [23].

4.3 The Modularity of Stereo Perception

The question of whether stereopsis is inextricably linked to other modes of visual perception, or whether it can stand alone as an independent mode, was clarified by the invention of the random-dot stereogram [15]. A random-dot stereogram consists of two synthetic images of random points, which are constructed to depict perspective views of the same virtual surface (Figure 6). A random dot stereogram can be constructed as follows. The points in, say, the left image are randomly placed. The right image consists of these same points, properly displaced to provide disparity values consistent with another view of some scene model. Each image by itself usually contains no depth information because it consists only of random dots. When the pair are viewed stereoscopically, however, the surface is readily perceived. Stereo vision is therefore capable of producing depth perception independently of any monocular cues such as size constancy, texture gradients, shading, and so on. Random-dot stereograms enable systematic comparison of human and machine performance because their parameters (such as noise, dot density, and the shape of the virtual surface) can be controlled precisely.

4.4 Relationships between Biological and Computational Models

So far, computational models of stereo vision have addressed only some of the many phenomena associated with human stereo perception. A complete computational theory would have to include descriptions and explanations of issues such as the limits to stereo discussed in Section 4.2, the relationship between fusion and diploptic depth perception, temporal effects (e.g., hysteresis), the relationship to accommodation and vergence, multistable and multivalued depth perception

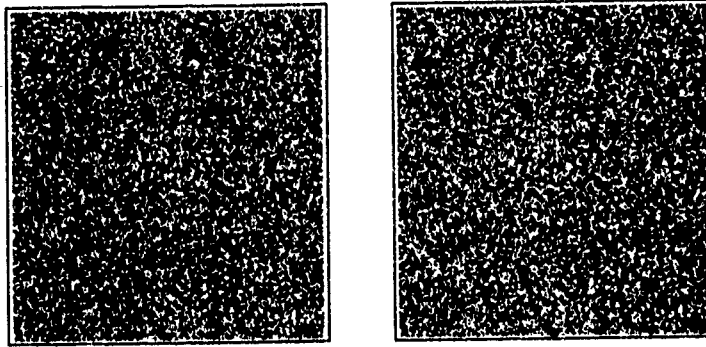


Figure 6: Random-dot stereogram.

(e.g., transparency), the relationship to monocular cues for depth, the nature of the features that are matched, and the perception of depth when there are rivalrous features (e.g. lines of different orientations). Nevertheless, in spite of the incomplete understanding of biological stereo vision, it is no accident that we find several analogies between computational stereo models and current knowledge of stereo vision in higher mammals. Historically, the formulation of models was often guided by biological knowledge (in particular, in the MPG model). From a pragmatic standpoint, machine models and biological neural systems operate under similar constraints, and this is increasingly so with the introduction of massively parallel architectures.

Independent spatial-frequency channels: The most immediate parallel is the use of resolution hierarchies. There is considerable evidence that the human visual system has at least four, and possibly more, independent channels tuned to different spatial frequencies separated by about one octave [27]. The lowpass pyramids of STEREOESYS, the bandpass sequences of MPG, and the bandpass pyramids of CYCLOPS perform the same function. The filters used in MPG and CYCLOPS, $\nabla^2 G$ and DOG, have about the same "Mexican hat" shape as the receptive fields of cells in the early visual system (see Figure 3).

Vertical disparity: Human stereo vision can deal with a small amount of vertical disparity, but it resorts to eye movements to align the visual fields as closely as possible [6,7]. The epipolar constraint allows one to use information about the camera orientation to remove vertical disparity. The MPG model assumes that this constraint is satisfied, while STEREOESYS and CYCLOPS use information about the camera orientation to apply it. Because error in the camera model may cause some residual vertical disparity, the search for corresponding points must be robust enough to handle at least a small amount of error. Apparently, the search-space reduction of the epipolar constraint is so useful in biological vision that vertical disparity is to be avoided as much as possible, but must be tolerated to degree.

Three pools of disparity. Studies of anomalous stereo vision suggest that there are three psychophysically distinct "pools" of disparity detectors, corresponding roughly to crossed (positive), uncrossed (negative), and near-zero disparity relative to the vergence point [24]. CYCLOPS uses the same representation, coupled with several octaves of hierarchy, to achieve efficiency without sacrificing dynamic range. At every level in the hierarchy, components of incremental disparity can assume only three local values: 1 (crossed), -1 (uncrossed), and zero. The base disparity, however, can grow by a factor of two across every level. The search space in any one level is therefore kept to a minimum, while the final composite disparity is allowed to have a substantial range.

Visual cortex. The visual cortex has a remarkably uniform structure.

Given what has been learned about the primary visual cortex, it is clear that one can consider an elementary piece of cortex to be a block about a millimeter square and two millimeters deep. To know the organization of this chunk of tissue is to know the organization for all of area 17; the whole must be mainly an iterated version of this elementary unit. [14]

As a computational substrate, the visual cortex is quite similar to a large grid of locally-connected processors. Since the cortical columns appear to be anatomically and functionally equivalent, they can be all be simulated by identical, locally interacting programs applied to different parts of the visual field, and therefore the visual cortex as a whole can be simulated efficiently by a fine-grained SIMD (single-instruction, multiple-data) architecture. All three systems described above have algorithmic structures that fit massively parallel SIMD architectures, and CYCLOPS is actually implemented on such a system.

5 Practical Applications

There are many existing and potential applications of stereo vision, but perhaps the two most important are cartography and robot vision. The current state of the art of computational stereo is most relevant to cartographic applications, where real-time performance is not required.

In standard cartographic practice, a human analyst uses a precise stereoscopic viewing and pointing device (called a *stereoanalytic plotter*) to develop a terrain-elevation map from stereo images. As one might imagine, this work is tedious, slow, and costly. Computational methods that automate stereopsis, sometimes implemented in special-purpose hardware or on massively parallel computers, are now practical for this important application.

The most important requirements of the cartographic application are accuracy and the ability to process very large images in a reasonable amount of time. Vertical stereo images are recorded from airplanes, although satellite-based images are now being used as well. Finely calibrated, large-format, long focal-length cameras are used, producing negatives of about 23 cm on a side, with about 60% overlap in field of view. The most common cartographic products are regular-grid terrain models, orthoimages (orthographic projections of the terrain), and contour plots. Using an absolute camera model, an automated cartographic system must convert raw disparity measurements into a dense, irregular mesh of matched-point coordinates, resample these points on a regular grid to produce a terrain-elevation model, and use the terrain model to produce a synthetic orthoimage.

Stereo vision would obviously be very useful for robots, whether as a straightforward mensuration device in a simple industrial robot or, at the other extreme, as part of an integrated perceptual system in an autonomous vehicle. The passive nature of stereo vision is highly desirable for some applications. Because these applications demand real-time performance, practical robot stereo vision awaits the development of faster, cheaper hardware. Continued improvement in the cost and performance of computer systems, is bringing this important application closer.

6 Open Questions

There has been substantial progress in developing computational models of stereo vision, but many open problems remain. With few exceptions, computational models have ignored such complications as transparent surfaces and occlusions. They typically produce a single-valued distance function, equivalent to a digital elevation map, sometimes with an identification of occluding contours. Such a simple representation cannot support

the functions of general purpose vision.

Computational models usually treat stereo as an independent module. While this is a useful reductionist tactic, stereo processing must ultimately be integrated into a more extensive and general perceptual (and possibly motor) framework. There is no reason why stereo processing should not be coupled with other perceptual modalities such as motion parallax and monocular perception of form, and with other cognitive functions such as memory and motor control. Human stereo vision is only one part of a complex ocular-motor system; effective stereo vision for advanced applications such as intelligent robots will doubtless require a similar degree of integration.

References

- [1] Barnard, S. T. and M. A. Fischler, Computational Stereo, *Computing Surveys*, Vol. 14, No. 4, 554-572, December 1982.
- [2] Barnard, S. T., Stochastic Stereo Matching over Scale, *International Journal of Computer Vision*, 3, 17-22, 1989.
- [3] Burt, P. and B. Julesz, "A disparity gradient limit for binocular fusion," *Science*, Vol. 208, 615-617, 1980.
- [4] Burt, P., "The Laplacian pyramid as a compact image code," *IEEE Trans. on Communications*, Vol. 31, 532-540, 1983.
- [5] Dhond, U. R. and J. K. Aggarwal, Structure from Stereo - A Review, *IEEE Trans. on Sys., Man, and Cyber.* Vol. 19, No. 6, November/December 1989.
- [6] Duwaer, A. L. and G. van den Brink, "What is the diplopia threshold?" *Vision Res.*, vol. 20, 295-309, 1981.
- [7] Duwaer, A. L. and G. van den Brink, "Diplopia thresholds and the initiation of vergence eye-movements," *Vision Res.*, vol. 21, 1727-1737, 1981.
- [8] Fender, D. and B. Julesz, "Extension of Panum's fusional area in binocularly stabilized vision, J. of the Opt. Soc. of Amer., Vol. 57, No. 6, 819-830, June 1967.
- [9] Fischler, M. A. and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *CACM*, Vol. 24, No. 6, 381-395, 1981.
- [10] Gennery, D. B., *Modeling the Environment of an Exploring Vehicle by Means of Stereo Vision*, Ph.D. Thesis, Stanford University, Computer Science Report STAN-CS-80-805, June 1980.

- [11] Grimson, W. E. L. G., "A computer implementation of a theory of human stereo vision," *Phil. Trans. Royal Soc. London*, Vol. B292, 217-253, 1981.
- [12] Grimson, W. E. L. G., "Computational experiments with a feature-based stereo algorithm," *IEEE Trans. Pattern Anal. and Mach. Intell.*, Vol. PAMI-7, No. 1, 17-34, Jan. 1985.
- [13] Hannah, M. J., A System for Digital Stereo Image Matching, *Photogrammetric Engineering and Remote Sensing*, Vol. 55, No. 12, 1765-1770, December 1989.
- [14] Hubel, D. H. and T. S. Weisel, Brain mechanisms of vision, in *the Brain*, W. H. Freeman and Co., New York, 1979.
- [15] Julesz, B., *Foundations of Cyclopean Perception*, Univ. of Chicago Press: Chicago, 1971.
- [16] Kandel, E. R. and J. H. Schwartz, *Principles of Neural Science*, 2nd ed., Elsevier Science Publishing Co., New York, 1985.
- [17] Kuffler, S. W., J. G. Nicholls, and A. R. Martin, *From Neuron to Brain*, 2nd ed., Sinaure Associates Inc., Sunderland, Massachusetts, 1984.
- [18] Longuet-Higgins, H. C., "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, 133-135, 1981.
- [19] Marr, D. and T. Poggio, "A computational theory of human stereo vision," *Proc. Royal Soc. London*, Vol. B207, 187-217, 1980.
- [20] Moravec, H. P., "Towards automatic visual obstacle avoidance," *Proc. 5th Int. Joint. Conf. Artificial Intell.*, p. 584, 1977.
- [21] Nagle, H. and B. Neumann, "On 3-D reconstruction from two perspective views," *Proc. IJCAI-81*, Aug. 1981.
- [22] Ogle, K. N., *Researches in Binocular Vision*, Saunders, Philadelphia, 1950.
- [23] Pollard, S. B., J. E. W. Mayhew, and J. P. Frisby, PMF: A stereo correspondence algorithm using a disparity gradient limit, *Perception*, Vol. 14, 449-470, 1985.
- [24] Richards, W., "Anomalous stereoscopic depth perception," *J. Opt. Soc. Amer.*, vol. 61, no. 3, 410-414, March 1971.
- [25] Tsai, R. Y. and T. S. Huang, "Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces," *IEEE PAMI*, vol. 6, no. 1, 13-27, Jan. 1984.
- [26] Trivedi, H. P. and S. A. Lloyd, The role of disparity gradient in stereo vision, *Perception*, Vol. 14, 685-690, 1985.
- [27] Wilson, H. R. and J. R. Bergen, "A four mechanism model for threshold spatial vision," *Vision Res.*, vol. 19, 19-32, 1979.

Recent Progress in CYCLOPS: A System for Stereo Cartography

Stephen T. Barnard

Artificial Intelligence Center
SRI International
333 Ravenswood Ave., Menlo Park, CA 94025

Abstract

Recent progress in the development of a system for automated cartography is presented. This system, called CYCLOPS, produces cartographic products — regular-grid elevation maps, ortho-images, and contour plots — from standard, digitized aerial stereo images. First, camera model information is used to produce corrected images with only horizontal parallax. The corrected images are then matched with a multigrid optimization algorithm. Essentially, the matching algorithm is a stochastic regularization method that tries to find the flattest dense disparity map that matches the photometry with least error. It does so by iterating a microcanonical version of simulated annealing across several levels of a resolution pyramid, using the results from the coarser levels to initialize the optimization search at the finer levels. After the corrected images are matched the disparity measurements are converted into a dense but irregular mesh of depth measurements, which is then resampled into a grid of elevations with respect to regularly spaced ground coordinates. A Bayesian interpretation of the method is given and some analogies between CYCLOPS and the human visual system are discussed.¹

1 Introduction

CYCLOPS is a system for automated cartography.² It began as a stochastic-optimization approach to stereo matching [1,2], but has recently evolved into a more complete system for cartographic terrain modeling, including software modules for camera modeling, epipolar resampling, and the generation of regular-grid elevation maps, ortho-images, contour plots, and synthetic perspective

views, in addition to the central task of image matching. One of the goals of this work has been to develop efficient stereo-processing methods for massively parallel SIMD architectures. The CYCLOPS system is implemented on the Connection Machine. The current implementation is capable of producing a dense terrain model (depth for every pixel) for a typical pair of 1024x1024 aerial stereo image in about eight minutes, using a Connection Machine with 4096 processors.

Historically, the computational modeling of stereo vision has been driven by two motivations. The practical applications of automated stereo are so important, especially in cartography and robotics, that many engineering-oriented approaches have been tried. These often use "correlation" techniques: patches of intensities in one image are searched for in the other image by maximizing a measure of correlation or minimizing a measure of error. The other motivation is the desire to model biological stereo, and these approaches are typically feature-based. discrete local features (usually edges) are matched across images. A third approach, which could be called the optimization approach, represents a dense disparity map as the state variable of a system, usually defined on a two-dimensional grid. Stereo matching is then formulated as an optimization problem: find the best disparity map by maximizing an objective function that measures the "quality" of the map, or, equivalently, minimizing an "energy" function that measures the lack of quality.

Optimization approaches to stereo usually involve minimizing some variant of the following function, the *stereo constraint equation*:

$$\mathcal{E} = \int \int ((\mathcal{L}(x - \frac{\mathcal{D}}{2}, y) - \mathcal{R}(x + \frac{\mathcal{D}}{2}, y))^2 + \lambda |\nabla \mathcal{D}|^2) u x dy, \quad (1)$$

where \mathcal{L} and \mathcal{R} are functions of image intensities (usually bandpassed images), $\mathcal{D} = \mathcal{D}(x, y)$ is a disparity map (a cyclopean map in this case), and λ is a regularization constant. The first term in the integrand is a measure of the photometric error associated with \mathcal{D} , and the second term (proportional to the squared magnitude of the gradient of disparity) is a measure of the first-order com-

¹The work described in this article was supported under DARPA contracts MDA903-86-C-0084, DACA76-85-C-0004, and 89F737300. Use of the Connection Machine was provided by DARPA.

²The name CYCLOPS derives from the notion of cyclopean disparity; that is, disparity defined symmetrically with respect to the two camera origins. In classical mythology a Cyclops is a member of a class of giants having one eye in the middle of the forehead.

plexity of \mathcal{D} . By minimizing \mathcal{E} one finds the flattest map with least photometric error, with the tradeoff between error and complexity specified by the regularization constant λ .

CYCLOPS uses a multigrid, stochastic-optimization algorithm to minimize a discrete version of equation (1). The following section describes the basic CYCLOPS algorithm for image matching. In particular, an important modification to the algorithm, the *three-pools mechanism*, is described. The three-pools mechanism not only leads to improved speed and generally better estimates, but it is also in agreement with psychophysical results. This leads to a further discussion of the analogies between CYCLOPS and current knowledge of biological vision. Next, the techniques used for completing the cartographic functionality of the system (generation of DTMs, ortho-images, etc.) are briefly described.

2 Basic CYCLOPS

2.1 Epipolar correction

Note that in equation (1) disparity is a scalar function: corresponding points between \mathcal{L} and \mathcal{R} are assumed to have the same y coordinate, so disparity can be characterized as a shift in x alone. Aerial stereo images are typically recorded with cameras pointed in approximately parallel directions and separated along a baseline approximately parallel to the cameras' x -coordinates. Ideally, there will be no y -parallax (also called vertical disparity), and the locations of corresponding points in the two images will differ only in x (this difference being called x -parallax or horizontal disparity). In practice, however, the camera coordinate systems will have some rotational error that will cause vertical disparity to be present.

The well-known *epipolar constraint* provides a way to use camera-model information to remove vertical disparity. Any point in three-dimensional space, together with the centers of projection of the two camera systems, defines a plane called an *epipolar plane*. The intersection of an epipolar plane with an image plane is called an *epipolar line*. Every point on an epipolar line in one image must correspond to a point on the corresponding epipolar line in the other image. The epipolar constraint therefore limits the searches for the matches to one-dimensional neighborhoods, as opposed to two-dimensional neighborhoods, with an enormous reduction in computational complexity.

CYCLOPS assumes that complete information about the camera models of the two stereo views is known. This model information includes, for each view, both the interior and exterior orientations. Interior orientation includes focal length, principal point, sampling rate, and possibly other parameters or even interpolation functions. Exterior orientation includes both the origin of

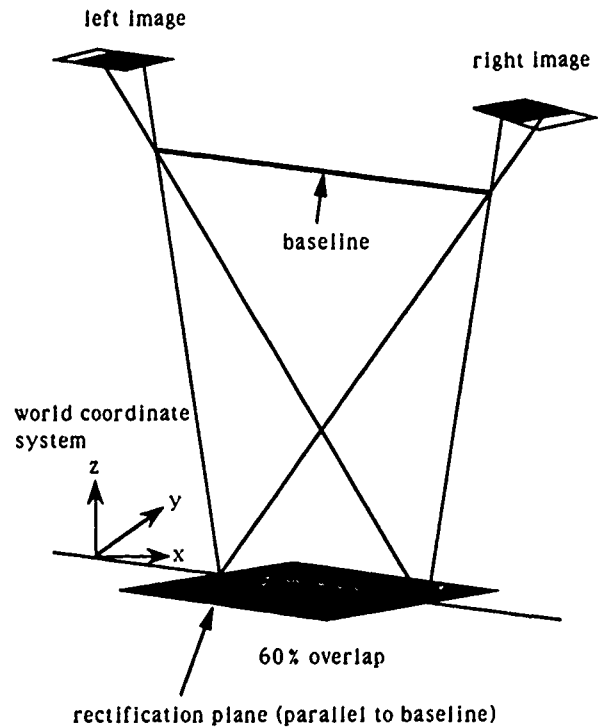


Figure 1: Epipolar correction.

the camera in an invariant world-coordinate system and an orthogonal transform matrix for translating between world and camera coordinates. This camera-model information is routinely compiled in standard cartographic practice. CYCLOPS uses the camera models to remove vertical disparity as shown in Figure 1. A rectification plane is selected parallel to the baseline, passing through the origin of world coordinates, and approximately parallel to the two image planes. (The normal of the rectification plane bisects the angle between the principal rays of the two views.) The images are then backprojected onto this plane (a perspective transform) and then resampled using bilinear interpolation. In typical aerial imagery the views are spaced to give approximately 60% overlap in coverage. Images are chosen in the overlapping area of the resampled images and given to the matching algorithm.

2.2 Bandpass pyramids

The multigrid strategy of CYCLOPS starts by converting each epipolar-corrected image into a bandpass pyramid, in which the image is separated into band-limited components separated by one octave. Furthermore, each component is sampled at a rate half that of the next higher-frequency one, producing a pyramid structure.

There is a particularly efficient method of computing this structure (see Burt [3]) that works as follows. Suppose we have a digital image $I(i, j)$ with dimensions

$2^N \times 2^N$. A low-pass, approximately Gaussian pyramid is defined recursively by

$$g_0(i, j) = I(i, j)$$

and

$$g_l(i, j) = \sum_{m=-2}^2 \sum_{n=-2}^2 w(m, n) g_{l-1}(i + m2^{l-1}, j + n2^{l-1})$$

for l up to $\log N$. The array $w(m, n)$ is a symmetric, separable generating kernel chosen to be approximately Gaussian. Note that g_l is a weighted average over a 5×5 pattern of samples in g_{l-1} , and that the sample spacing, 2^{l-1} , doubles across each level. Next, bandpass components are constructed as approximate differences-of-Gaussians:

$$DOG_l(i, j) = g_l(i, j) - g_{l+1}(i, j), l = 0, \dots, (\log N) - 1.$$

This structure is sometimes called a Laplacian pyramid because it provides a good approximation to the Laplacian of the Gaussian kernel, $\nabla^2 G$.

There are several benefits in using the Laplacian pyramid:

1. Disparity has a linear scaling property: if the sampling rate of the images is reduced by a factor of 2, then the range of disparity is also reduced by a factor of 2. Therefore, at some very coarse sampling rate, disparity is negligible. A matching system can start at a low level of resolution, find a coarse disparity relatively quickly, and then use it to initialize the search at the next higher octave, etc.
2. Raw image intensities are not usually invariant across stereo images for a variety of reasons, including sensor bias and illumination effects. However, these systematic sources of error primarily affect the low-frequency components of the image. Using bandpassed images provides a kind of automatic gain control, similar to the effect of lateral inhibition in neural systems.
3. It may be impossible to eliminate vertical disparity completely. That is, the epipolar constraint may be satisfied only approximately. Nevertheless, y -parallax will always be much smaller than x -parallax, typically only 2 or 3 pixels at the highest resolution. At coarser resolutions it will be negligible. The resolution pyramid therefore allows a degree of tolerance to vertical disparity.

CYCLOPS constructs truncated Laplacian pyramids from each epipolar-corrected image:

$$\{(L_l, R_l) | l = 0, \dots, l_{max}\}.$$

The lowest level of resolution, l_{max} , is determined by the size of the original images and by a limit on the

range of disparity between them. For typical aerial images with which we work the lowest resolution in the pyramids would be 32×32 . If the original images were 1024×1024 , the pyramids would be six levels deep and the system would accommodate a disparity range of 127 pixels.

2.3 The three-pools mechanism

CYCLOPS uses a version of simulated annealing to minimize a discrete version of the stereo constraint equation at each level of resolution:

$$E_l = \sum_i \sum_j (L_l(i - \lfloor \frac{D_l}{2} \rfloor, j) - R_l(i + \lfloor \frac{D_l}{2} \rfloor, j))^2 + \sum_i \sum_j \lambda |\nabla D_l(i, j)|^2 \quad (2)$$

where L_l and R_l are components of the Laplacian pyramids of the left and right images, D_l is a discrete disparity map, and the operator $|\nabla \cdot|^2$ is now a discrete approximation of the squared gradient magnitude computed over four nearest neighbors.

The discrete disparity map, D , can be represented within the system in different ways. In earlier versions of CYCLOPS, D was represented as a grid of 8-bit signed integers and could attain any values (within this range) at any level. In the current system, however, a representation called the *three-pools mechanism* is used to obtain more efficiency without sacrificing dynamic range. It works as follows: At each level of the resolution pyramid the disparity map D is the sum of a constant *base* disparity (inherited from the previous level) and a variable *incremental* disparity:

$$D_l(i, j) = D_l^0(i, j) + \delta_l(i, j)$$

with $\delta_l(i, j) \in \{-1, 0, 1\}$. Now δ_l is the state variable of the system and is represented as a signed 2-bit integer. The local photometric errors (the terms in the first sum in equation (2)) are tabulated into the three possible values they can attain at each level. Note that incremental disparity is interpreted as a correction to the coarse answer obtained from the previous level. In this way the system can handle a substantial range of disparity, but the search space at each level of the pyramid is kept as small as possible.

The three-pools mechanism is illustrated in Figure 2. The optimization algorithm assigns disparities at the coarsest level as shown by the filled-in circles in 2(a). It then sets the base disparity (the shaded circles) at the next higher level 2(b) of the resolution hierarchy as specified by the following equation:

$$D_l^0(i, j) = 2D_{l+1}(\lfloor i/2 \rfloor, \lfloor j/2 \rfloor).$$

At this level the incremental disparity is allowed to vary between -1 and 1 during the next optimization phase, as

indicated by the short arrows in the figure. After the optimization determines the best incremental disparity, it uses the composite disparity in 2(c), $D_i^0(i, j) + \delta_i(i, j)$, to set the base disparity of the next level.

Note that if the difference in composite disparity between neighboring grid points becomes too large the points become decoupled. This is caused by a threshold in the squared-gradient-magnitude operator:

$$|\nabla D(i, j)|^2 = \sum_{\{i', j'\} \in \mathcal{N}(i, j)} \min(1, (D(i, j) - D(i', j'))^2)$$

where $\mathcal{N}(i, j)$ specifies the four nearest neighbors of $\{i, j\}$.³ Once a component of the squared disparity gradient becomes greater than 1, further separation of the disparity values between neighboring points has no effect on the second term in equation (2). The effect of this decoupling is that the system can establish sharper estimates for disparity along occlusion boundaries. Occlusions are not usually encountered in aerial cartographic images — at least not at the level of detail present in the data — but they can be significant in oblique, ground level stereo.

2.4 Stochastic optimization

The method for minimizing E at each level, called microcanonical annealing, is somewhat different from the standard forms of simulated annealing. It has been adapted from a result in statistical physics [4] to be particularly efficient on data-parallel computers with no floating-point hardware. Perhaps the most interesting property of microcanonical annealing is that, unlike the standard forms, it doesn't use temperature as a control parameter. Instead, it represents kinetic energy explicitly with a grid of demons. Temperature is *measured* as a statistical feature of the system (the average demon energy). Details can be found in [1].

Like conventional simulated annealing methods, the microcanonical algorithm generates a sequence of states, $\{S_k\}$, that in the limit converges to a Boltzman distribution:

$$Pr(S) \propto e^{-E_S/\beta},$$

where β is temperature (in units of energy), which characterizes the rate of decrease of the probability of a state with increasing energy. Simulated annealing works by gradually reducing β ,⁴ causing the distribution of states to "relax" from a high-temperature, highly uncertain ensemble into a minimum-energy state at $\beta = 0$. In practice, the algorithm does not find the absolute minimum, but rather a good approximation to the minimum with high probability. The major advantage of simulated annealing over deterministic gradient-descent methods is

³A value of $\lambda = 64$ seems to work well for this operator, assuming that the images are digitized properly into 256 grey levels.

⁴Or, in the case of microcanonical annealing, reducing kinetic energy.

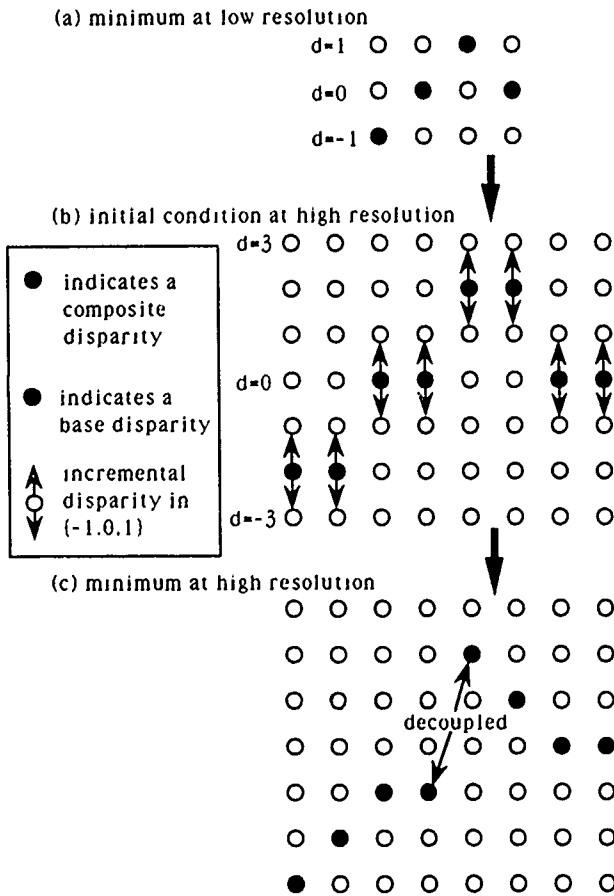


Figure 2: The three-pools mechanism.

that, by permitting "uphill" moves to states of higher energy, it can avoid getting trapped in locally stable states.

Subpixel estimates of disparity can be obtained by averaging over many states. When the finest level of resolution has been obtained the system is not cooled to $\beta = 0$; instead, it is heated to a fairly high temperature and a sequence of disparity maps is generated. The average disparity (over this sequence) quickly converges to a good subpixel estimate. In terms of the discussion to follow, this technique averages over samples from the posterior distribution.

2.5 A Bayesian interpretation

One view of the computation is that of maximum *a posteriori* (MAP) estimation [7]. We observe (L, R) which contains information about D . The Bayesian approach is to find D that maximizes $Pr\{D|(L, R)\}$. Applying Bayes' Rule we have

$$Pr\{D|(L, R)\} = \frac{Pr\{(L, R)|D\}Pr\{D\}}{Pr\{(L, R)\}}$$

The probability in the denominator, $Pr\{(L, R)\}$, is constant, so the posterior probability $Pr\{D|(L, R)\}$ is proportional to the numerator:

$$Pr\{D|(L, R)\} \propto Pr\{(L, R)|D\}Pr\{D\}.$$

$Pr\{(L, R)|D\}$ is determined by the photometric error in the stereo constraint equation. (Given a disparity map, image pairs with less error are more likely.) The prior probability $Pr\{D\}$ is determined by the variational term. (Disparity maps that are flatter are more likely.)

The state variable of CYCLOPS, that is to say the discrete cyclopean disparity map D , is an MRF whose distribution of energies is guaranteed to be Boltzman in the limit. Writing the probabilities in terms of the distributions we have

$$\begin{aligned} Pr\{D|(L, R)\} &\propto \left(e^{-E_1(L, R, D)/\beta}\right) \left(e^{-E_2(D)/\beta}\right) \\ &\propto e^{-(E_1 + E_2)/\beta} \\ &\propto e^{-E/\beta} \end{aligned}$$

where E_1 and E_2 are the error and the variational terms in E , respectively. Therefore, Bayes' Rule tells us that by minimizing E we are maximizing the posterior probability $Pr\{D|(L, R)\}$.

3 Analogies to Biological Stereo

The design of CYCLOPS was intended to perform dense stereo matching as efficiently as possible on massively-parallel, locally-connected hardware; there was no conscious intent to model biological stereo vision. Nevertheless, there are several nontrivial analogies to be drawn

between this computational model and current knowledge of the working of the human visual system. CYCLOPS is by no means unique among computational models in this respect. The MPG model, in particular, which was expressly based on neurobiological and psychophysical evidence [8,9,10], shares many of the features of CYCLOPS.

3.1 Independent spatial-frequency channels

There is considerable evidence that the human visual system has at least four, and possibly more, independent channels tuned to different spatial frequencies separated by about one octave [13]. The Laplacian pyramid in CYCLOPS and similar structures in other models play the same role. The reasons why this encoding is important in CYCLOPS (efficiency of coarse-to-fine search, automatic gain control, and less sensitivity to y -parallax) suggest functions these channels may perform in human vision.

3.2 Sensitivity to vertical disparity

Human stereo vision can deal with a small amount of vertical disparity but it resorts to eye movements to align the visual fields as closely as possible [5,6]. CYCLOPS uses information about the camera orientation to remove vertical disparity. Error in the camera model may cause some residual vertical disparity, but the simulated-annealing search is robust enough to handle this under reasonable conditions. Apparently, the search-space reduction of the epipolar constraint is so useful that vertical disparity is to be avoided as much as possible, but must be tolerated to degree, in both CYCLOPS and the human visual system.

3.3 Three pools of disparity

Studies of anomalous stereo vision suggest that there are three psychophysically distinct "pools" of disparity detectors, corresponding roughly to crossed (positive), uncrossed (negative), and near-zero disparity relative to the vergence point [12]. CYCLOPS uses the same representation, coupled with several octaves of resolution hierarchy, to achieve efficiency without sacrificing dynamic range. At every level in the hierarchy, components of incremental disparity can assume only three local values: 1 (crossed), -1 (uncrossed), and zero. The base disparity, however, can grow by a factor of two across every level. The search space in any one level is therefore kept to a minimum, while the final composite disparity is allowed to have a substantial range.

3.4 Visual cortex

The first stereo computation in the human visual system occurs in the primary visual cortex. This part of

the brain is remarkably uniform across its surface: a column of tissue about a millimeter square appears to characterize an elementary building block [11]. As a computational substrate, the visual cortex is then quite similar to a large grid of locally-connected processors. Since the cortical columns appear to be anatomically and functionally equivalent, they can be all be simulated by identical, locally interacting programs applied to different parts of the visual field, and therefore the visual cortex as a whole can be simulated efficiently by a fine-grained SIMD (single-instruction, multiple-data) architecture such as the Connection Machine.

4 Depth Determination

Once the disparities of the epipolar-corrected images are determined it is a relatively simple matter to compute the depth of each matched point. Using the camera model, rays from the centers of projection through the matched points are intersected to produce a grid of tuples in world coordinates:

$$\{x(i, j), y(i, j), z(i, j)\} .$$

Unfortunately, these tuples define an irregular mesh of points, while what is normally required in cartography is a grid of elevations at regular intervals of ground coordinates:

$$\{z(x, y) | x = x_0 + m\Delta x, y = y_0 + n\Delta y\} .$$

This is a standard resampling problem, equivalent to scan conversion in computer graphics. It is accomplished by triangulating the mesh of irregular points and then, in parallel, resampling each triangle on the regular grid. At the same time as the regular-grid elevation map is produced, the system creates a synthetic orthographic image of the terrain. CYCLOPS can also generate synthetic perspective views of the scene by texture mapping the ortho-image onto the grid of elevations and rendering the scene using a 3D graphics package developed on the Connection Machine at SRI.

5 Future Work

CYCLOPS must be implemented on less costly hardware if it is to be a cost-effective tool for cartography. Even a 4K Connection Machine, which is the smallest configuration available, costs more than \$500,000. Fortunately, we can expect massively data-parallel systems to be much cheaper in the near future. Today one can buy a system for less than \$200,000 that could run the CYCLOPS algorithm considerably faster than a 4K Connection Machine.

The development of much faster hardware will open the door to a new stereo application: robot vision. Stereo obviously has much potential for robotics,

whether as a straightforward mensuration tool for industrial applications or as part of an integrated perceptual system of an autonomous, mobile robot.

The mobile robot application, in particular, poses several challenging problems not encountered in the cartographic domain. If stereo vision is to be used effectively for real-time control, it must respond at nearly video frame rates. Instead of processing independent pairs of images, the system would have to integrate temporally a continuous stream of stereo data. The stereo subsystem would have to be coordinated with and integrated into the total perceptual, motor, and cognitive framework. Of course, if the robot could control the stereo cameras quickly and accurately this could be used to advantage. The data would be streams of oblique, ground-level images of general terrain. One would expect much greater ranges of disparity, the presence of occlusions, and in general a more complex scene than one finds in cartography.

References

- [1] Barnard, S. T., "Stochastic stereo matching over scale," *International Journal of Computer Vision*, 3, 17-22 (1989).
- [2] Barnard, S. T., "Stochastic stereo matching on the Connection Machine," *Proc. Image Understanding Workshop*, May 1989.
- [3] Burt, P., "The Laplacian pyramid as a compact image code," *IEEE Trans. on Communications*, Vol. 31, 532-540, 1983.
- [4] M. Creutz, "Microcanonical Monte Carlo simulation," *Phys. Rev. Lett.* 50, 1411-1414, 1983.
- [5] Duwaer, A. L. and G. van den Brink, "What is the diplopia threshold?" *Vision Res.*, vol. 20, 295-309, 1981.
- [6] Duwaer, A. L. and G. van den Brink, "Diplopia thresholds and the initiation of vergence eye-movements," *Vision Res.*, vol. 21, 1727-1737, 1981.
- [7] Geman, S. and D. Geman, "Stochastic relaxation, Gibbs distributions, and Bayesian restoration of images," *IEEE Trans. PAMI*, vol. 6, 721-741, 1984.
- [8] D. Marr and T. Poggio, "A theory of human stereo vision," *Proc. Roy. Soc. London*, vol. B 204, 301-328, 1979.
- [9] Grimson, W. E. L. G., "A computer implementation of a theory of human stereo vision," *Phil. Trans. Royal Soc. London*, Vol. B292, 217-253, 1981.
- [10] Grimson, W. E. L. G., "Computational experiments with a feature-based stereo algorithm," *IEEE*

Trans. Pattern Anal. and Mach. Intell., Vol. PAMI-7, No. 1, 17-34, Jan. 1985.

- [11] Hubel, D. H. and T. S. Weisel, Brain mechanisms of vision, in *the Brain*, W. H. Freeman and Co., New York, 1979.
- [12] Richards, W., "Anomalous stereoscopic depth perception," *J. Opt. Soc. Amer.*, vol. 61, no. 3, 410-414, March 1971.
- [13] Wilson, H. R. and J. R. Bergen, "A four mechanism model for threshold spatial vision," *Vision Res.*, vol. 19, 19-32, 1979.

A Context-Based Recognition System for Natural Scenes and Complex Domains

Thomas M. Strat and Martin A. Fischler
Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, California 94025

Abstract

This paper describes progress in an ongoing project concerned with recognizing objects in complex scene domains, and in particular, the domain that includes the natural outdoor world. Traditional machine recognition paradigms assume either (1) that all objects of interest are definable by a relatively small number of explicit shape models, or (2) that all objects of interest have characteristic, locally measurable features. The failure of both assumptions in a complex domain such as the natural outdoor world has a dramatic impact on the form of an acceptable architecture for an object recognition system.

In our work, we make the use of contextual information a central issue, and explicitly design a system to identify and use context as an integral part of recognition. In so doing, we provide a new paradigm for visual recognition that eliminates the traditional dependence on stored geometric models and universal image partitioning algorithms. Initial experimentation with the system on ground-level outdoor imagery has already demonstrated competence beyond what we believe is attainable with other existing vision systems¹.

1 Introduction

Much of the progress that has been made to date in machine vision has been based, almost exclusively, on shape comparison and classification employing locally measurable attributes of the imaged objects (e.g., color and texture) [2, 4, 9]. Natural objects viewed under realistic conditions do not have uniform shapes that can be matched against stored prototypes, and their local surface properties are too variable to be unique determiners of identity. The standard machine vision recognition paradigms fail to provide a means for reliably recognizing *any* of the object classes common to the natural outdoor world (e.g.,

trees, bushes, rocks, and rivers). In this paper and its predecessors [5, 6, 15], we have outlined a new paradigm which explicitly invokes context and stored knowledge to control the complexity of the decision-making processes involved in correctly identifying natural objects and describing natural scenes.

Scene description is properly viewed as a problem in scientific discovery. Ultimately a collection of assertions must be provided, each assertion stating the identity and relevant attributes (e.g., spatial location) of some object depicted (or possibly invisible, but inferred to be present) in an imaged scene. There are two critical differences between the problem domain addressed in this paper and the class of problems capable of being solved by existing machine vision paradigms.

1.1 Hypothesis Generation

The first critical difference is that of hypothesis generation. The acceptance of either of two assumptions trivializes the hypothesis generation problem for conventional machine vision systems. Conventional systems have no effective machinery for hypothesis generation when both assumptions are invalid — in a sense, this failure of both assumptions is one of the main attributes of a complex domain.

The two assumptions are:

1. All objects of interest are defined by a relatively small number of explicit shape models. This makes it computationally feasible to exhaustively search for the presence of these models (via "geometric alignment") as a way of producing a suitable description of some given scene (as in [3] and [11], for example)
2. All objects of interest have characteristic features, homogeneous and locally measurable in an image (e.g., color or texture), which are reliable indicators of the object's identity. This either allows direct determination of the presence of objects using statistical decision theoretic methods (based on classification of the corresponding feature vectors); or

¹Supported by the Defense Advanced Research Projects Agency under contracts MDA903-86-C-0084, DACA76-85-C-0004 and 89F737300.

permits the successful employment of a "universal" partitioning algorithm which finds regions (homogeneous in the given attributes) in the imagery corresponding to the objects of interest.

The validity of assumption (2) allows a single universal procedure (the partitioning algorithm) to find regions in the imagery which are good delineations of the objects of interest. These regions provide an effective basis for generating the required hypotheses for object location and identity.

The failure of both assumptions has a dramatic impact on the form of an acceptable architecture/control-structure for an object recognition system. Not only are we required to introduce an explicit mechanism for computationally feasible hypothesis generation, but we must provide additional machinery for representing and accessing the supporting information necessary for such hypothesis generation. We are forced to cross the line from what has been called model-based vision to an "AI-complete" problem domain.

Much of our work has addressed devising the representations and control structures (i.e., the introduction of context sets in a production rule type framework) needed to merge the vision-specific and more general AI technologies.

1.2 Hypothesis Evaluation

The second critical difference between the approach we propose and existing machine vision paradigms is their distinct treatment of the scene objects to be recognized. Conventional machine vision paradigms define scene objects to be independent entities which can (and should be) isolated from the rest of the scene and then labeled on the basis of their differences from other objects for which we have names (and models). The system we describe in this paper, called Condor, treats objects as component parts of larger contexts (many different contexts for each object) from which they cannot be separated — like quarks in modern physics, they never appear in isolation and have no independent existence. Once a context is recognized, its individual components may be instantiated and given names.

The need to embed objects in more extensive contexts, rather than treating them as independent entities, is due to the following considerations (in complex domains):

1. The image appearance of an object can be quite variable, not only due to intrinsic shape variability, but also due to viewing conditions (e.g., resolution, occlusion, and lighting). The object's relationship to its surroundings is often a major factor in determining its identity — even for a human.
2. Some objects (such as a river or a bridge) cannot be defined, let alone recognized in an image, independent of their embedding in the surrounding terrain.

The architectural and computational implications of context-based definitions is of equal significance to those caused by the need to provide special machinery for hypothesis generation. Fortunately, much of the needed machinery can serve both functions. Thus, context sets not only define objects, but control the generation of (candidate) hypotheses and the subsequent evaluation of those hypotheses.

Having to deal with contexts, rather than independent objects introduces a major increase in computational complexity. Contexts are much more numerous than the objects they are composed of, and contexts are less precisely defined. The verification problem changes from identifying objects based on sufficient conditions (e.g., of similarity) to that of eliminating alternatives based on failure to satisfy necessary conditions. We are required to deduce much more about the nature of the overall scene — especially its physical structure.

To the extent that the Condor architecture, and its representations and partitioning of knowledge are successful in advancing the state-of-the-art in machine vision for complex natural scenes, we believe that this success will also contribute to increased competence in other non-vision related AI classification tasks in complex domains, but especially to those tasks which require decision making involving both iconic and symbolic information.

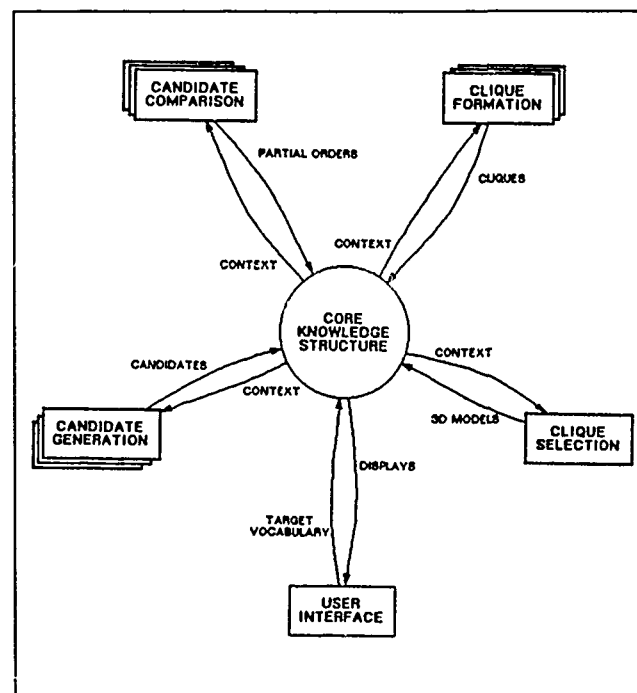


Figure 1: Conceptual architecture of Condor

2 Conceptual Architecture

The conceptual architecture of the system we describe, called Condor (for context-driven object recognition), is depicted in Figure 1. The input to the system is an image or set of images that may include intensity, range, color, or other data modalities. The primary output of the system is a labeled 3D model of the scene. The labels included in the output description denote object *classes* that the system has been tasked to recognize, plus others from the recognition vocabulary that happen to be found useful during the recognition process. An object *class* is a category of scene features such as

{sky, ground, geometric-horizon, skyline, foliage, bush, tree-trunk, tree-crown, tree, trail, ... }

A central component of the architecture is a special-purpose knowledge/database used for storing and providing access to knowledge about the visual world, as well as tentative conclusions derived during operation of the system. In Condor, these capabilities are provided by the Core Knowledge Structure (CKS).

The conceptual architecture is much like that of a production system; there are many computational processes interacting through a shared data structure. Interpretation of an image involves the following four process types.

- Candidate generation (hypothesis generation)
- Candidate comparison (hypothesis evaluation)
- Clique formation (grouping mutually consistent hypotheses)
- Clique selection (selection of a "best" description)

Each process acts like a daemon, watching over the knowledge base and invoking itself when its contextual requirements are satisfied. All processing occurs asynchronously and each process is assumed to have access to sufficient computational resources. All processes have access to the entire knowledge base, but each type of process will only store the kind of information shown in the diagram (Figure 1).

2.1 Context sets

The invocation of all processing operations in Condor is governed by context through the use of *context sets*: an action is initiated only when one or more of its controlling context sets is satisfied. Thus, the actual sequence of computations, and the labeling decisions that are made, are dictated by contextual information (stored in the Core Knowledge Structure), by the computational state of the system, and by the image data available for interpretation.

A *context set* is a collection of context elements that are sufficient for inferring some relation or carrying out

some operation on an image. Syntactically, a context set is embedded in a rule denoted by

$$L : \{CE_1, CE_2, \dots, CE_n\} \Rightarrow A$$

where L is the name of the class associated with the context set, A is an action to be performed, and the CE_i comprise a set of conditions that define a context. For convenience, we often refer to the entire rule as a context set.

Example: The context set {SKY-IS-CLEAR, CAMERA-IS-HORIZONTAL, RGB-IS-AVAILABLE} defines a set of conditions under which it is appropriate to use the operator BLUE-REGIONS to delineate candidate sky hypotheses.

There is a collection of context sets for every class in the recognition vocabulary. In theory, Condor performs the actions A that are associated with every *satisfied* context set.

A context set is satisfied only when the known context is sufficient to establish the truth of all its elements. Often it will not be possible to establish whether a context-set element is true or false, in which case the element is considered to be unsatisfied.

Visual interpretation knowledge is encoded in context sets, which serve as the uniform knowledge representation scheme used throughout the system. Context sets are employed in three varieties of rules.

- Type I: Candidate generation
- Type II: Candidate evaluation
- Type III: Consistency determination

Context sets of each type are constructed for each object class in the recognition vocabulary. The most difficult part of building any AI system is encoding the knowledge that drives the system. Constructing context sets in Condor is tantamount to knowledge base construction and remains a critical task requiring a solid understanding of the limitations and applicability conditions of potential image understanding routines. Condor has been designed with this in mind, and offers several features that facilitate this process.

First, the construction task is eased somewhat by the separation of the knowledge base according to classes. Therefore, when constructing context sets for class L , the only other classes that must be considered are those that are immediately relevant for recognizing instances of class L .

Second, context sets need only define sufficient conditions for applying the associated operation — they need not attempt to define the full boundary of applicability. Thus, one can be quite conservative when constructing context sets, only encoding knowledge that is clearly relevant, and ignoring that which may be dubious.

Third, although it is desirable that the context sets and their associated operations be as infallible as possible, they need not be perfect. The entire architecture of Condor has been designed to achieve reliable recognition results, even in the presence of unreliable operators, imperfect evaluators, and faulty decision-makers. This is achieved primarily through the use of large numbers of redundant operations in every stage of processing, so that a single mistake is unlikely to affect the final interpretation.

Finally, we have proposed a mechanism whereby context sets can be modified automatically, using the experiences of the system to refine the knowledge base incrementally. The collection of context sets can be allowed to evolve, with or without human intervention. Some form of learning is essential if a large system with a broad range of competence is to be constructed.

2.2 Hypothesis generation

The customary approach to recognition in machine vision is to design an analysis technique that is competent in as many contexts as possible. In contrast to this tendency toward large, monolithic procedures, the strategy embodied in Condor is to make use of a large number of relatively simple procedures. Each procedure is competent only in some restricted context, but collectively, these procedures offer the potential to recognize a feature in a wide range of contexts. The key to making this strategy work is to use contextual information to predict which procedures are likely to yield desirable results, and which are not.

While it may be extremely difficult to write a recognition procedure that is competent across many different contexts, it is often quite easy to devise a procedure that works well in some specific context. For example, finding foliage that is silhouetted against the sky is far simpler than finding foliage in general. Similarly, finding foliage in an environment where only a single species of tree occurs, is easier than finding foliage associated with any kind of tree. By assembling a collection of such context-specific procedures, it has been possible to recognize foliage in many different situations under a wide variety of conditions.

A collection of recognition procedures is associated with each class in the recognition vocabulary. Of course, no procedure, not even one applied in very restricted contexts, will be sufficiently reliable that its results can be accepted with confidence. Accordingly, the output of each procedure is treated as a candidate hypothesis.

A *candidate hypothesis* is any image feature that is potentially an instance of some specified class. In most of our examples, an image region is associated with each candidate, but in general, a candidate is any hypothesis that asserts the presence of some object in the 3D scene depicted in the image being analyzed. Every candidate

is associated with the class of which it is potentially an instance.

A large portion of the Condor architecture is devoted to sorting out the better candidate hypotheses from the poorer ones. Figure 2 shows the generation and subsequent processing of candidates throughout the system.

The invocation of recognition procedures is governed by candidate generation context sets which define the conditions under which it is sensible to employ each recognition procedure.

Example: Horizontal surface patches are likely to be part of the ground, but they can only be computed when range data is available. The context set in the following production controls the invocation of the associated operator.

GROUND :

{CLIQUE-IS-EMPTY, DENSE-RANGE-IS-AVAILABLE }
⇒ HORIZONTAL-SURFACE-PATCHES

The elements in a candidate generator context set encode the assumptions that were made when the associated operator was written. This formalism ensures that each operator will only be employed in circumstances in which it can reasonably be expected to succeed. The context set not only identifies an applicable procedure, but also supplies the information to establish intelligently the inevitable parameters (such as a threshold or a window-size) associated with that operator.

Obviously, context sets can be very specific, very generic, or anywhere in between. It is intended that candidate generator context sets be provided that span this range. One encodes very specific context sets for operators that work well only in very special circumstances, presumably a context that has some special significance to the larger goals of the embedded system. Generic operators that provide reasonable performance over a broad range of contexts, are employed when the more competent specialized procedures are not applicable. Generally, the more candidate generator context sets that are provided, the more operators that will be applicable in any given context. Ideally, there will always be multiple operators invoked so that the system need never rely on a single routine.

It should be clear that it is possible to make use of large, carefully constructed procedures when they exist. Thus, if one has already expended a great deal of effort tuning a large, monolithic recognition procedure, it can be incorporated into Condor alongside any other operators that might also exist.

The interaction of context sets across classes is of interest. The context elements in one context set may refer to the existence of other labeled entities. For example, a tree trunk candidate generation routine may require knowledge of the ground location as part of its context. Whenever a need for recognition of other classes is detected, Condor adds that class to its list of labels that

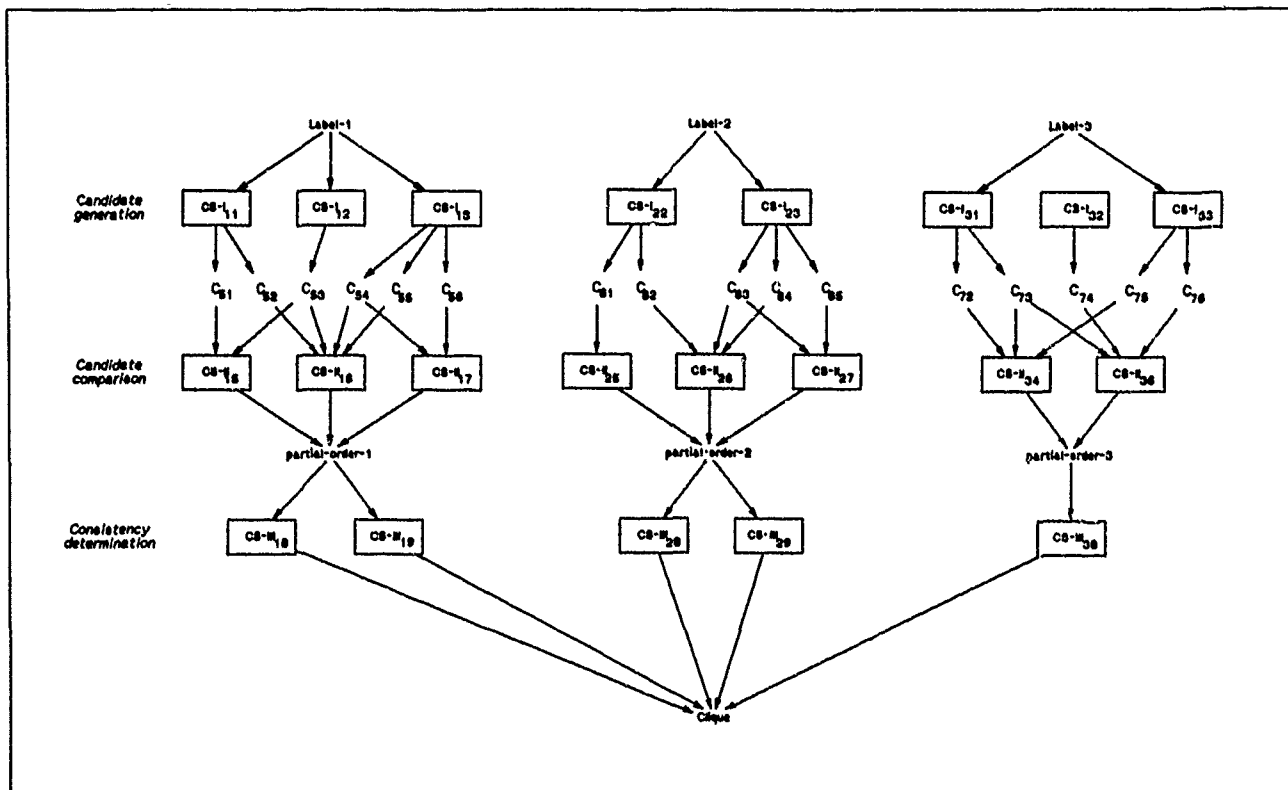


Figure 2. Schematic diagram of data dependency in Condor. A separate organization of information is required for each clique being formed.

are actively being recognized. In this way, when Condor is tasked to recognize a specific class from its target vocabulary, it will automatically seek to instantiate other classes from its recognition vocabulary that are relevant.

2.3 Clique formation

The result of the candidate generation process is a collection of candidates for each label in the active recognition vocabulary. Because the operators cannot be expected to be sufficiently robust, extra steps must be taken to find those candidates that truly are instances of their associated classes.

To obtain this increase in reliability, we make use of a principle of maximal coherency which holds that the best interpretation of an image is the one which coherently explains the greatest portion of a scene. Candidates that are not consistent with a partial image interpretation cannot be part of the final interpretation. The goal is to find a mutually consistent set of candidates that explains as much of the image as possible.

A set of mutually consistent candidate hypotheses, called a *clique*, represents a possible interpretation of the image. Condor builds a number of cliques and chooses the "best" one as its final interpretation. Naturally, it would be computationally infeasible to generate all possible cliques — instead, cliques are generated in a special

order (described in the next section) to increase the likelihood that a good interpretation is found early. Thus, the longer that Condor analyses an image, the better its interpretation is likely to be.

Inconsistency is determined by context-specific procedures whose application is mediated by context sets (see also Figure 2).

Example: A candidate for ground cannot extend above the skyline.

GROUND : { CLIQUE-CONTAINS(skyline) }
 \Rightarrow PARTIALLY-ABOVE-SKYLINE

As was the case with candidate generation, the inconsistency determination routines are assembled into context sets that encode the assumptions necessary for their successful application. Each operator tests a candidate for consistency with all the incumbents already present in a clique. If any satisfied context set finds a candidate to be inconsistent, then it is not admitted into that clique, although it may participate in other cliques. Thus, consistency determination context sets provide necessary (but not sufficient) conditions for clique inclusion.

A clique contains a collection of candidates annotated with inferred 3D properties and relations. The inconsistency operators encode geometric and physical relationships that must be consistent with known facts about

the environment and the various semantic classes. The operators may involve either 2-D image-plane computations or such 3D constraints as size, support, orientation, occupancy of solid objects, etc. The 2-D constraints are useful for rapidly eliminating some candidates when they are easily seen to be inconsistent, or when sufficient 3D information cannot be established to allow more sophisticated spatial reasoning procedures to be applied. The consistency determination context sets include context elements that specify what 3D information must be known. Their use causes an attempt to infer that information if it is not already known.

2.4 Candidate comparison

The search for the largest coherent set of candidates can be combinatorially infeasible without further constraint — the number of potential cliques is exponential in the number of candidates. For this reason, cliques are generated in a special order.

At any point during the processing of an image, there will be a collection of candidates for each label to be instantiated. Some of these candidates are obviously better examples of the class denoted by the label than are others. By first building cliques from the best candidates of each class, we are much more likely to encounter good cliques early in the search (typically several within the first half-dozen cliques). Condor has used this "best-first" strategy to successfully avoid the combinatorics that would otherwise prevent recognition.

The task here is to order the candidates within each class so the better ones may be added to cliques before the others. The difficulty is choosing a suitable metric to accomplish this ordering. For most classes of interest in the outdoor world, there is no single evaluation metric that gives a reliable ordering. One could conceive of multiple metrics that evaluate the candidates along various dimensions, but that would still leave the problem of comparing multi-dimensional evaluation vectors. In order to justify a weighted sum of the vector components, one would have to make the unlikely assumption of some form of independence. A similar independence assumption would be required if the evaluation measures were to be given a probabilistic interpretation and combined using probability theory.

The solution we have adopted is to make use of multiple evaluators, but not to assume that they are independent in any way. Instead, they are used to compare two candidates for a given label, with each evaluator casting a vote for the candidate it ranks higher. If all evaluators favor one candidate over another, a preference ordering between the candidates is established. Otherwise, no ordering is imposed. The net effect of comparing (pairwise) all candidates for a given label is to impose a partial order on those candidates. The candidates at the tops of the partial orders will be tested for consistency with the

cliques before those below them.

An *evaluator* is a function that scores the relative likelihood that a candidate for a class is actually an instance of that class. The evaluators that apply in any context are described by *candidate evaluation context sets*.

Example: When viewed obliquely, the ground usually exhibits a horizontally striated texture. HORIZONTALLY-STRiated is a function that measures this property within a candidate region.

```
GROUND : {CAMERA-IS-HORIZONTAL}
⇒ HORIZONTALLY-STRiated
```

As before, the context sets allow the relevant knowledge to be subdivided into manageable pieces. The elements of each context set encode the conditions under which a relatively simple-minded evaluation function gives meaningful information. It is intended that many evaluation functions be provided within context sets, so that robust comparisons result whenever a unanimous vote occurs. One candidate is preferred over another only when all evaluators occurring in satisfied context sets score it as least as high as the other candidate.

As always, context-set elements that refer to other object classes cause other computations to be triggered. Satisfied context-set elements also provide information for setting parameters that may be required by the associated evaluation functions.

The structure of the comparisons is noteworthy because it contrasts with the way comparisons are performed in nearly every other recognition system. The usual approach is to partition an image and to consider which of several potential class labels is the best description of a region. In Condor, we start with several partitions (candidates) and consider which of several candidates is the most likely instance of a class. For example, a conventional recognition system would consider whether a particular region was more likely to be a tree trunk or a road. Condor would have several potential delineations of a tree trunk and would consider which is the best description of the trunk.

This departs from conventional approaches in two significant ways. First, comparing candidate regions for a given label requires knowledge of the semantics of that label only, whereas the customary approach of comparing two labels for a given region requires knowledge of the relationships between many semantic categories. When considering which candidate is the best tree trunk, Condor needs to know only about tree trunks and related categories (such as branches, roots, and the ground). In contrast, to decide what label to assign to a given region using a conventional approach, one must be able to compare any pair of labels. This requires knowledge of the relationships between every pair of semantic categories, and grows rapidly as new classes are added to the recognition vocabulary. The Condor orientation provides a

basis for believing that sufficient knowledge might eventually be encoded in the system to allow robust comparison even in a large-scale system.

Second, we enforce the condition that the comparisons lead to a preference only if one candidate is clearly a better choice than the other. With this conservative approach, we can reap additional computational savings by pruning large portions of the search for maximally consistent cliques. For example, if candidate C_1 is clearly a better instance of class L than candidate C_2 in the context of a particular clique, and C_1 is found to be inconsistent with that clique, then C_2 can be eliminated as a potential member of that clique as well. Ruling out C_2 may eliminate other candidates recursively. Thus we avoid the need to test the consistency of C_2 and any of its inferiors. Furthermore, it may at times be impossible otherwise to establish C_2 as inconsistent, in which case this pruning step prevents the clique from being contaminated with a bad candidate. Although it does not follow logically that C_2 cannot be a class L instance, its elimination is a powerful heuristic that is nearly always justified. We can afford to take this chance because additional cliques will be generated simultaneously that may happen to avoid repeating an unjustified elimination. Thus even when some generators yield unreliable candidates, and the comparisons make occasional mistakes, it may still be possible to build a clique that yields a completely accurate semantic labeling of an image.

2.5 The recognition process

Let us summarize the processing steps that have been described so far (Figure 2). For each label in the active recognition vocabulary, all candidate generation context sets are evaluated. The operators associated with those that are satisfied are executed, producing candidates for each class. Candidate comparison context sets that are satisfied are then used to evaluate each candidate for a given class, and if all such evaluators prefer one candidate over another, a preference ordering is established between them. These preference relations are assembled to form partial orders over the candidates, one partial order for each class. Next, a search for mutually coherent sets of candidates is conducted by incrementally building cliques of consistent candidates, beginning with empty cliques. A candidate is nominated for inclusion into a clique by choosing one of the candidates at the top of one of the partial orders. Consistency determination context sets that are satisfied are used to test the consistency of a nominee with candidates already in the clique. A consistent nominee is added to the clique, an inconsistent one is removed from further consideration with that clique. Further candidates are added to the cliques until none remain. Additional cliques are generated in a similar fashion as computational resources permit. Ultimately, one clique is selected as the best se-

mantic labeling of the image on the basis of the portion of the image that is explained and the reliability of the operators that contributed to the clique.

Each of the processing steps occurs simultaneously in our conceptual view, but there are some implicit sequencing constraints. Candidate evaluators begin to construct partial orders as soon as candidates become available. Incremental addition of candidates to cliques begins as soon as partial orders are available. Theoretically, there is no need to wait for one stage to complete before latter stages are begun, but it may be desirable when computational resources are limited.

The interaction among context sets is significant. The addition of a candidate to a clique may provide context that could trigger a previously unsatisfied context set to generate new candidates or establish new preference orderings. For example, once one bush has been recognized, it is a good idea to look specifically for similar bushes in the image. A candidate generation context set that includes an element that is satisfied only when a bush is in a clique implements this tactic.

Similarly, as cliques evolve, the partial orders for each class may change. Ideally, one should wait for all candidate generation and comparison activity to subside before nominating a candidate into a clique. We regard this synchronization as an implementation issue that is not of theoretical importance because additional cliques will always be generated later.

It is important to remember that multiple cliques will be in various stages of construction simultaneously. Each clique has its own partial orders from which to choose, although many candidates will be identical in several or all of the cliques. Context set satisfaction is determined individually for each clique.

3 Implementation of Condor

3.1 Processing Sequence

All of the computations carried out by Condor are controlled by context sets. At any given time, there might be many satisfied context sets whose operators could be invoked. Condor, as implemented, evaluates context sets in an order that is designed to provide additional information rapidly. For example, it is sensible to build all partial orders as completely as possible before starting to build cliques, although this is not required by the conceptual architecture. Although the context sets are evaluated in a fixed order, their satisfaction depends on the context so far derived. Thus, the order in which operators are invoked depends primarily on the contextual information. The order of context set evaluation we have chosen serves mainly to accelerate the interpretation of images.

The sequence of operation in Condor is summarized in Figure 3. The serialization of an inherently paral-

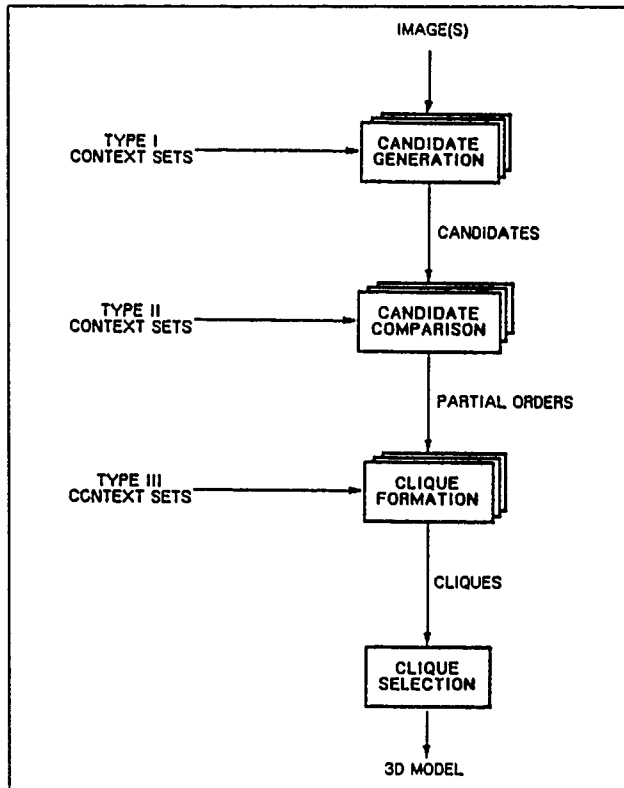


Figure 3: Sequence of computation

Architecture is complicated by the interdependencies among the processing steps. When first presented with an image and tasked to recognize a target vocabulary, Condor generates candidates and compares them to impose a partial order on the candidates in the target vocabulary. Any additional classes that are found to be of use are added to the active recognition vocabulary and are processed similarly. Next, a candidate from the top of one of the partial orders is added to a clique. This changes the context relevant to that clique, so the candidate generation process is repeated and the partial orders are reevaluated in that new context. A comprehensive caching mechanism is employed to prevent reevaluating any operations that have not changed. A new nominee is chosen from the tops of the partial orders and checked for consistency with the clique. If it is found to be consistent, it is added to the clique and removed from its partial order. If inconsistent, it is removed from further consideration for membership in that clique, although it may join another clique later. The inconsistent nominee is removed from its partial order along with any candidate over which it is preferred. This cycle is repeated until no candidates remain for nomination, thus completing the development of the first clique.

Additional cliques are generated by iterating the entire process. Any operations that occurred before construction of the first clique began need not be repeated since their context is still valid. To accomplish this, the sys-

tem is rewound to that point and construction of the second clique begins. Condor generates different cliques by nominating candidates in different orders. Many strategies exist for selecting different orders and the heuristic nominating function can be modified to implement them. The strategy that Condor standardly uses is to seed each clique with a candidate that had been ruled out by an earlier clique, thereby guaranteeing that a new and different clique will result.

After each clique is completed, it is compared with the best previous clique to determine which interpretation of the image is better. There is no theoretically sound way of comparing two cliques, and the method we employ is somewhat ad hoc. Each clique is scored on the basis of the portion of the image that is explained and the reliability of the operators that generated the candidates in the clique. The higher scoring clique is retained and additional cliques are generated until a scoring threshold is exceeded or available computation time is exhausted. At that point, the highest scoring clique is accepted as the best interpretation of the image, and the candidates it contains are considered to have been recognized.

The contents of this best clique are then used to update the 3D model of the environment. Newly found objects are inserted in the CKS. Candidates depicting previously known objects are used to update the location, size, shape, and appearance of that object in the CKS. The name of the operator that successfully delineated each object in the image is stored with the object so that it might be invoked again when that object comes in the field of view. The result is an updated model of the visual world, that will provide more context for the recognition of objects in subsequent images.

3.2 Representation of context

Because Condor has been designed to make use of a persistent store of information about the visual world, it is necessary to provide a mechanism for its representation. Condor requires access to scene objects based on their location or any of various semantic properties. This role is filled by the Core Knowledge Structure.

The CKS is an object-oriented knowledge/database that was originally designed to serve as the central information manager for a perceptual system [13, 14]. The following four facilities of the CKS are of particular importance for Condor.

3.2.1 Multiple Resolution

The CKS employs a multiresolution octree to locate objects only as precisely as warranted by the data. Similarly, a collection of geometric modeling primitives are available to represent objects at an appropriate level of detail. In parallel with the octree for spatial resolution is a semantic network that represents things at multiple levels of semantic resolution. Condor's recognition

vocabulary is represented as nodes in the semantic network, which allows the system to refer to objects at an appropriate level in the abstraction hierarchy.

3.2.2 Inheritance and inference

The CKS uses the semantic network to perform some limited types of inference that ease the burden of querying the data store. Thus, query responses are assembled not only from those objects that syntactically match the query, but also from objects that can be inferred to match given the relations encoded in the semantic network. For example, the CKS can be queried for all trees within 10 meters of any dirt road, and will find all such trees regardless of whether they were originally categorized as oaks or pines or whether any roadway was present when they were instantiated in the database. Spatial inference is provided based on geometric constraints computed by the octree manipulation routines. Inheritance of attributes that are unspecified is performed in a similar fashion. For example, a query for all objects taller than 5 meters will be satisfied by all trees not specifically known to be shorter than 5 meters, but not satisfied by any rocks (unless they are known to be higher than 5 meters).

3.2.3 Conflicting data

One of the realities of analyzing imagery of the real world is that conflicts will result from mistakes in interpretation and from unnoticed changes in the world. The database used by Condor must not collapse when conflicting information is stored. The CKS treats all incoming data as the opinions of the data sources, so logical inconsistencies will not corrupt the database. Similarly, values derived through multiple inheritance paths are treated as multiple opinions. This strategy has several advantages and disadvantages. Rather than fusing information as it arises, the CKS has the option of postponing combination until its results are needed. This means that the fusion can be performed on the basis of additional information that may become available, and in a manner that depends on the immediate task at hand. Some information may never be needed, in which case the CKS may forego its combination entirely. The disadvantages are the need to store a larger quantity of data and a slowed response at retrieval time. For an object recognition system like Condor, the CKS seems to provide the right tradeoff.

Condor uses the multiple opinion facility to store the attributes of recognized objects. Each attribute value is annotated with the image in which it was identified, its time of acquisition, and time of recognition. In so doing, it is possible to reason about the validity of the stored data, and to react accordingly. The opinion mechanism is also used to store multiple cliques in Condor. Each

candidate is stored in the CKS as the opinion of the clique to which it pertains.

3.2.4 User interface

Although Condor is designed to be a fully automated recognition system, a comprehensive user interface is invaluable for development and debugging. The CKS provides a menu-driven query mechanism that is useful for inspecting the intermediate states of computation. In addition, the CKS has been integrated with SRI's Cartographic Modeling Environment [10] to provide a capability of generating synthetic views of terrain. This allows one to visualize the contents of the database from an arbitrary viewpoint by rendering a synthetic image. Doing so provides a window into the information that Condor is assuming as it interprets an image.

3.3 Context set construction

Context sets are the key to any recognition abilities that Condor demonstrates. While we have not yet evolved a precise procedure for designing context sets, we can provide some insight based on our experience in building context sets for natural object recognition.

Type I context sets (candidate generation) are constructed based on an assessment of what operators may work for each label in the recognition vocabulary. Based on a representative sample of imagery from the target domain, we composed image processing operations that work reasonably well in various circumstances. Factors that influenced the choice of which operators to include were the likelihood of success, the ease of implementation, the lack of any alternative operators, and the availability of existing code. Table 1 lists the types of operators that are actually employed by Condor to generate candidates (for the experimental site in the Stanford foothills). For each operator, the assumptions that it requires are encoded as context elements in a context set that controls the invocation of the operator. These context elements limit the situations in which the operator will be applied, ensure the existence of any required data, and establish the parameter settings associated with the operator.

Type II context sets (candidate evaluation) are assembled from evaluation metrics that can be used to compare two candidates. Context elements that define the conditions under which the metrics are meaningful are collected into context sets for each label in the recognition vocabulary. The metrics themselves need not order candidates perfectly, but should perform substantially better than a random ordering. Condor requires a unanimous vote of all applicable metrics before ordering two candidates, so a faulty metric is likely to leave some candidates unordered but not reverse ordered. It is important that preferences be correct when they are made. Non-preferences will require more cliques to be searched

Algorithm	Explanation
ASSOCIATION	Finds connected sets of pixels in a binary image
STRIATIONS	Finds the orientation and strength of local texture
DELINEATION	Finds line-like structure
OUTLINING	Finds the boundary of a region
THRESHOLDING	Uses scale-space techniques to choose thresholds
EDGE FINDING	Any of several well-known edge-finding routines
CONTRAST enhancement	Stretches the histogram of an image
SMOOTHING	Low-pass filter
HISTOGRAMMING	Computes a histogram and associated statistics
TEXTURE	Any of several well-known algorithms for measuring texture
SEGMENTATION	Completely partitions an image using KNIFE [12]
DENSE STEREO	Computes a dense depth image using CYCLOPS [1]
SPARSE STEREO	Computes depths at some easily correlated points [8]
HOMOGENEITY	A noise tolerant algorithm for measuring local homogeneity

Table 1: Candidate generation operators

Evaluation metric	Explanation
ABOVE-GEOMETRIC-HORIZON	Raised objects are more likely found above the horizon
ABOVE-SKYLINE	Raised objects above the skyline are preferred
BELOW-GEOMETRIC-HORIZON	Prefer ground candidates below the horizon
BELOW-SKYLINE	Prefer ground candidates below the skyline
BLUE	Prefer blue sky candidates on a sunny day
BRIGHT	Prefer bright sky candidates
ELLIPSOIDAL	When range data is available, prefer ellipsoidal bushes and tree-crowns
ELLIPTIC	Prefer bushes and tree-crowns that are shaped like ellipses (in 2D)
GREEN	Prefer green grass in the winter and spring in California
HIGHLY-TEXTURED	Prefer foliage candidates that are highly textured
HORIZONTAL	Prefer ground candidates that are horizontal (in 3D)
HORIZONTALLY-STRIATED	Prefer ground candidates that exhibit horizontal striations
NEAR-TOP	Prefer sky candidates that are near the top of the image
NO-SKY-BELOW	Prefer bush and rock candidates that are not above the sky
REASONABLE-SIZE	Prefer trees and bushes that are sized appropriately
SIMILAR-COLOR	Prefer candidates that are similar in color to known objects
SIMILAR-TEXTURE	Prefer candidates that have similar texture as a known object
UNDEFINED-RANGE	Prefer sky candidates that is uncorrelated in stereo
2D-VERTICALITY	Prefer tree trunks that are approximately vertical in the image
3D-VERTICALITY	When range is available, prefer tree trunks that are vertical

Table 2: Evaluation metrics

Consistency constraint	Explanation
ABOVE-SKY-REGION	Most objects must not be completely off the ground
LEANING	Objects that lean too much are unsupported
MISMATCHED-BRIGHTNESS	The intensity of sky, for example, cannot vary too much
NOT-SUPPORTED-BY-GROUND	Most plants must be rooted in the ground
OVERLAPS-IN-IMAGE	Some hypotheses that are inconsistent in 2D are ruled out
PARTIALLY-ABOVE-SKYLINE	The ground cannot extend above the skyline
PARTIALLY-BELOW-GEOMETRIC-HORIZON	The sky cannot extend below the horizon

Table 3: Consistency constraints

#	Class	Context elements	Operator
1	SKY	CLIQUE-IS-EMPTY	SEGMENTATION-REGIONS
2	SKY	CLIQUE-IS-EMPTY	WEAKLY-TEXTURED-REGIONS
3	SKY	CLIQUE-IS-EMPTY	WEAKLY-STRIATED-REGIONS
4	SKY	CLIQUE-IS-EMPTY	BRIGHT-REGIONS
5	SKY	CLIQUE-IS-EMPTY \wedge SKY-IS-CLEAR \wedge RGB-IS-AVAILABLE	BLUE-REGIONS
6	SKY	LAST-SELECTED-CANDIDATE-IS(sky)	SIMILAR-REGIONS
7	GROUND	CLIQUE-IS-EMPTY	SEGMENTATION-REGIONS
8	GROUND	CLIQUE-IS-EMPTY \wedge CAMERA-IS-HORIZONTAL	HORIZONTAL-STRIATION-REGIONS
9	GROUND	CLIQUE-IS-EMPTY \wedge DENSE-RANGE-IS-AVAILABLE	HORIZONTAL-SURFACE-PATCHES
10	GROUND	LAST-SELECTED-CANDIDATE-IS(ground)	SIMILAR-REGIONS-REGIONS
11	FOLIAGE	CLIQUE-IS-EMPTY	TEXTURE-ABOVE-THRESHOLD
12	FOLIAGE	CLIQUE-IS-EMPTY	VEGETATIVE-TRANSPARENCY
13	FOLIAGE	CLIQUE-IS-EMPTY \wedge RGB-IS-AVAILABLE	GREEN-REGIONS
14	FOLIAGE	LAST-SELECTED-CANDIDATE-IS(foliage)	SIMILAR-REGIONS
15	FOLIAGE	CLIQUE-IS-EMPTY \wedge DENSE-RANGE-IS-AVAILABLE	HIGHLY-FRACTAL-REGIONS
16	RAISED-OBJECT	CLIQUE-IS-EMPTY	SEGMENTATION-REGIONS
17	RAISED-OBJECT	CLIQUE-IS-EMPTY	VERTICAL-STRIATION-REGIONS
18	RAISED-OBJECT	CLIQUE-IS-EMPTY \wedge DENSE-RANGE-IS-AVAILABLE	DENSE-REGIONS-ABOVE-GROUND
19	RAISED-OBJECT	CLIQUE-IS-EMPTY \wedge SPARSE-RANGE-IS-AVAILABLE	SPARSE-REGIONS-ABOVE-GROUND
20	RAISED-OBJECT	LAST-SELECTED-CANDIDATE-IS(complete-sky)	NON-SKY-REGIONS-ABOVE-SKYLINE
21	COMPLETE-GROUND	LAST-SELECTED-CANDIDATE-IS(geometric-horizon)	REGION-BELOW-GEOMETRIC-HORIZON
22	COMPLETE-GROUND	LAST-SELECTED-CANDIDATE-IS(ground)	UNION-OF-GROUND-REGIONS
23	COMPLETE-GROUND	LAST-SELECTED-CANDIDATE-IS(skyline)	REGION-BELOW-SKYLINE
25	COMPLETE-SKY	LAST-SELECTED-CANDIDATE-IS(sky) \wedge SITE-IS(Stanford-hills)	UNION-OF-SKY-REGIONS

Table 4: Type I Context Sets: Candidate Generation

#	Class	Context elements	Operator
41	SKY	ALWAYS	ABOVE-HORIZON
42	SKY	SKY-IS-CLEAR \wedge TIME-IS-DAY	BRIGHT
43	SKY	SKY-IS-CLEAR \wedge TIME-IS-DAY	UNTEXTURED
44	SKY	SKY-IS-CLEAR \wedge TIME-IS-DAY \wedge RGB-IS-AVAILABLE	BLUE
45	SKY	SKY-IS-OVERCAST \wedge TIME-IS-DAY	BRIGHT
46	SKY	SKY-IS-OVERCAST \wedge TIME-IS-DAY	UNTEXTURED
47	SKY	SKY-IS-OVERCAST \wedge TIME-IS-DAY \wedge RGB-IS-AVAILABLE	WHITE
48	SKY	SPARSE-RANGE-IS-AVAILABLE	SPARSE-RANGE-IS-UNDEFINED
49	SKY	CAMERA-IS-HORIZONTAL	NEAR-TOP
50	SKY	CAMERA-IS-HORIZONTAL \wedge CLIQUE-CONTAINS(complete-sky)	ABOVE-SKYLINE
51	SKY	CLIQUE-CONTAINS(sky)	SIMILAR-INTENSITY
52	SKY	CLIQUE-CONTAINS(sky)	SIMILAR-TEXTURE
53	SKY	RGB-IS-AVAILABLE \wedge CLIQUE-CONTAINS(sky)	SIMILAR-COLOR
61	GROUND	CAMERA-IS-HORIZONTAL	HORIZONTALLY-STRIATED
62	GROUND	CAMERA-IS-HORIZONTAL	NEAR-BOTTOM
63	GROUND	SPARSE-RANGE-IS-AVAILABLE	SPARSE-RANGES-FORM-HORIZONTAL-SURFACE
64	GROUND	DENSE-RANGE-IS-AVAILABLE	DENSE-RANGES-FORM-HORIZONTAL-SURFACE
65	GROUND	CAMERA-IS-HORIZONTAL \wedge CLIQUE-CONTAINS(complete-ground)	BELOW-SKYLINE
66	GROUND	CAMERA-IS-HORIZONTAL \wedge CLIQUE-CONTAINS(geometric-horizon) \wedge \neg CLIQUE-CONTAINS(skyline)	BELOW-GEOMETRIC-HORIZON
67	GROUND	TIME-IS-DAY	DARK
71	FOLIAGE	ALWAYS	HIGHLY-TEXTURED
72	FOLIAGE	ALWAYS	HIGH-VEGETATIVE-TRANSPARENCY
73	FOLIAGE	CAMERA-IS-HORIZONTAL	NEAR-TOP
74	FOLIAGE	RGB-IS-AVAILABLE	GREEN
76	RAISED-OBJECT	SPARSE-RANGE-IS-AVAILABLE	SPARSE-HEIGHT-ABOVE-GROUND
77	RAISED-OBJECT	DENSE-RANGE-IS-AVAILABLE	DENSE-HEIGHT-ABOVE-GROUND
78	RAISED-OBJECT	CAMERA-IS-HORIZONTAL \wedge CLIQUE-CONTAINS(complete-sky)	ABOVE-SKYLINE

Table 5: Type II Context Sets: Candidate Evaluation

#	Class	Context elements	Operator
81	SKY	GEOMETRIC-HORIZON-KNOWN	PARTIALLY-BELOW-GEOMETRIC-HORIZON
82	SKY	ADDING-TO-CLIQUE	INCONSISTENT-WITH-CLIQUE
83	SKY	ADDING-TO-CLIQUE \wedge CLIQUE-CONTAINS(sky)	MISMATCHED-BRIGHTNESS
84	SKY	SPARSE-RANGE-IS-AVAILABLE	MUST-NOT-HAVE-FINITE-RANGE
87	GROUND	CLIQUE-CONTAINS(complete-sky)	PARTIALLY-ABOVE-SKYLINE
88	GROUND	ADDING-TO-CLIQUE	INCONSISTENT-WITH-CLIQUE
89	GROUND	DENSE-RANGE-IS-AVAILABLE	SLOPE-TOO-STEEP
91	FOLIAGE	ADDING-TO-CLIQUE	INCONSISTENT-WITH-CLIQUE
93	COMPLETE-GROUND	ADDING-TO-CLIQUE	INCONSISTENT-WITH-CLIQUE

Table 6: Type III Context Sets: Consistency Determination

but will not lead to incomplete recognition results. Table 2 shows some of the evaluation metrics that are used by Condor.

Type III context sets (consistency determination) define the conditions under which inconsistency of a candidate with a clique can be established. Any constraints that make it impossible for a candidate hypothesis to be valid given the assumption that the candidates already in the clique are correct, are encoded and assembled into Type III context sets. It is important that inconsistent candidates be correctly identified so that physically impossible cliques are not constructed. However, it is not necessary that a complete definition of consistent candidates be encoded. This asymmetry was designed specifically because it is far simpler to specify what could not be a tree, for example, than it is to specify what is a tree. Some of the consistency determination constraints that are used by Condor are listed in Table 3.

4 Example of natural object recognition



Figure 4. A typical image from the Stanford foothills

To illustrate the basic processing sequence, Condor was tasked to recognize the sky, the ground, and the foliage, appearing in the image shown in Figure 4. This relatively easy image was acquired in the foothills behind the Stanford University campus in the afternoon of a sunny day using an ordinary 35mm camera. To make the description as clear as possible, some of the machinery incorporated in Condor has been deactivated while creating this example. In particular, no prior knowledge of the terrain or features on that terrain is used. The digitized image is a single monochrome 8-bit frame, no

color, stereo, or other range data is used.

4.1 Candidate generation

Condor begins by generating candidates for each of the classes in the target vocabulary. The relevant candidate generation context sets are shown in Table 4. Tables 5 and 6 show the relevant Type II and Type III context sets used in this example.

While generating candidates for the sky label, context set 5 was not satisfied because no color image is available and context set 6 was not satisfied because no candidates have been selected yet for inclusion in a clique. Context sets 1-4 are satisfied and the sky candidates they generate are shown in Figure 5a. Notice that three of the candidates (910, 912, and 914) are fairly similar — Condor must eventually sort out which one(s) to include in each clique based on how well they fit in the context of other members in the clique.

Ground candidates are generated by context sets 7-10 and are shown in Figure 5b. Foliage candidates are generated by context sets 11-15. The candidate generation context sets for raised-object are used to generate additional foliage candidates because foliage is a subcategory of raised-object in the abstraction hierarchy. The foliage candidates are depicted in Figure 5c.

4.2 Candidate comparison

Next, Condor compares the candidates for each class to construct the partial orders. Candidate evaluation context sets 41-53 are used for evaluating sky candidates. Only context sets 41, 42, 43, and 49 are satisfied. Their associated operators are used to evaluate each of the sky candidates and the results are assembled in Table 7. Each evaluator returns a score between 0.0 and 1.0. Only the relative magnitude of this score for each evaluator is meaningful. The scores are not normalized across evaluators because there is no basis to do so.

Evaluator	909	910	911	912	914
ABOVE-HORIZON	1.00	1.00	1.00	1.00	1.00
BRIGHT	0.44	0.71	0.94	0.76	0.67
UNTEXTURED	0.19	0.67	0.52	0.50	0.36
NEAR-TOP	0.51	0.79	0.37	0.73	0.66

Table 7: Initial evaluation of sky candidates

Examining the table reveals that candidate 910 was scored at least as high as candidate 909 by every evaluator. Therefore, 910 is preferred over 909 as a sky candidate. Other unanimous preferences are

910 \succ 914, 912 \succ 909, 912 \succ 914, and 914 \succ 909.

These relations are assembled into a partial order and displayed in Figure 6, after removing transitivity. Candidate 909, which roughly delineates the trees, is at the

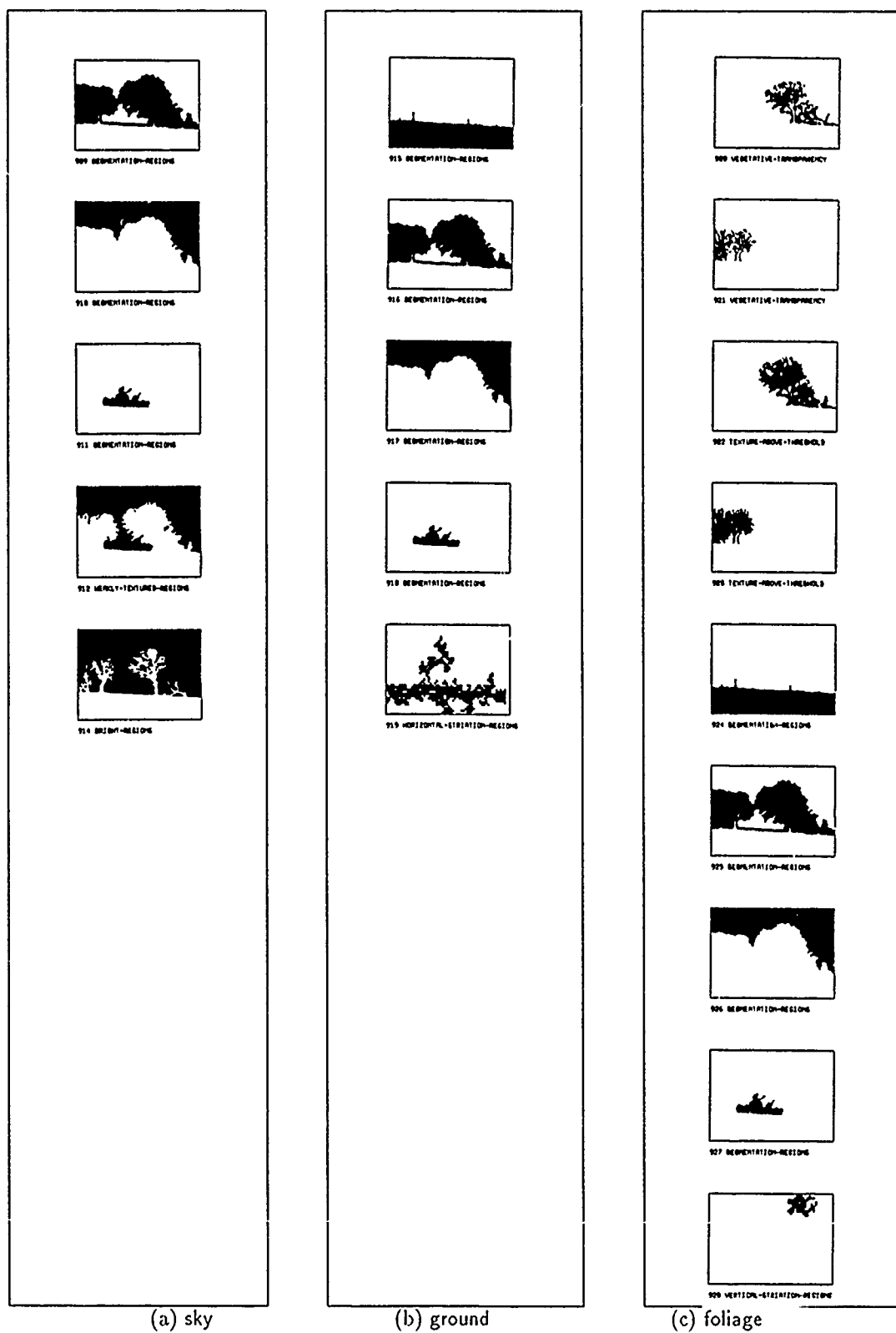


Figure 5: Some candidates generated by Condor

bottom of the partial order as one would hope. Candidates 910, 911, and 912, were found to be equally promising sky regions.

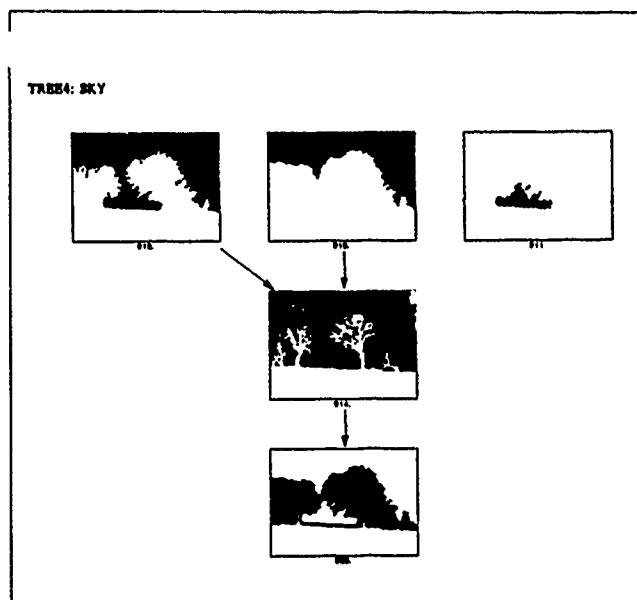


Figure 6: Partial order of candidates for sky

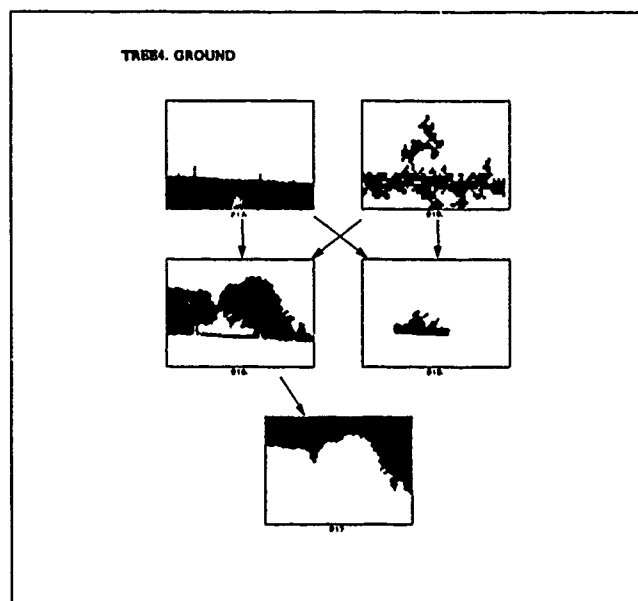


Figure 7: Partial order of candidates for ground

The partial orders generated for ground and foliage are shown in Figures 7 and 8.

At this point the active recognition vocabulary is {sky, ground, foliage, geometric-horizon, complete-sky, complete-ground, skyline} and Condor proceeds to generate candidates and partial orders for the remainder of these classes².

²These are of no special interest and are not shown.

4.3 Clique formation

When all satisfied context sets for classes in the active recognition vocabulary have been employed, Condor begins to build cliques of mutually consistent candidates. The candidates at the tops of the four partial orders are eligible to be introduced into an (empty) clique. The choice of which candidate to nominate first is made with the aid of a heuristic that chooses on the basis of the reliability of the operator that generated the candidate, the desirability of adding the candidate's class to the clique, the nearness of the candidate to the camera, and the size of the candidate. If this choice is made poorly, it may lead to a small clique and more cliques will have to be generated before a large, mutually coherent clique is constructed.

According to the heuristic, the geometric-horizon candidate is chosen first and added as the sole candidate in Clique 1. This tentative conclusion constitutes new context, albeit for Clique 1 only. All Type I context sets are reevaluated to see if any new candidates are generated, and all Type II context sets are reevaluated to update the partial orders. The only new candidate that is produced is a complete-ground candidate generated by context set 21. Type II context set 66 is now satisfied and adds BELOW-GEOMETRIC-HORIZON to the list of evaluators for ground candidates. Its use happens to cause no changes in the ground partial order.

Condor continues to test candidates for inclusion in the clique, adding those that are consistent and pruning those that are not. After each addition to the clique, the context sets are reevaluated to determine whether additional candidate generation operators or evaluation metrics have become applicable in the new context. The partial orders are updated after each change and processing continues until no candidates remain to be tested.

Figure 9 shows the complete sequence of nominations to the first clique. The composite labeling of the image that results from those that were accepted is given by Figure 10. A total of 36 candidates were generated for this clique, of which 18 were accepted in the clique, 10 were found to be inconsistent, and 8 were pruned without testing.

4.4 Clique selection

In this case, the first clique generated did a good job recognizing the target vocabulary, but Condor has no definitive way of knowing this. Condor generates additional cliques to see if its interpretation can be improved. In this experiment, six additional cliques were generated, but none of them exceeded the reliability and coverage of the components of the first clique.

As a result, Condor selects the first clique as its best interpretation and stores its results in the CKS database to be used as context for future reference. When range data is available, it is used to position the objects in

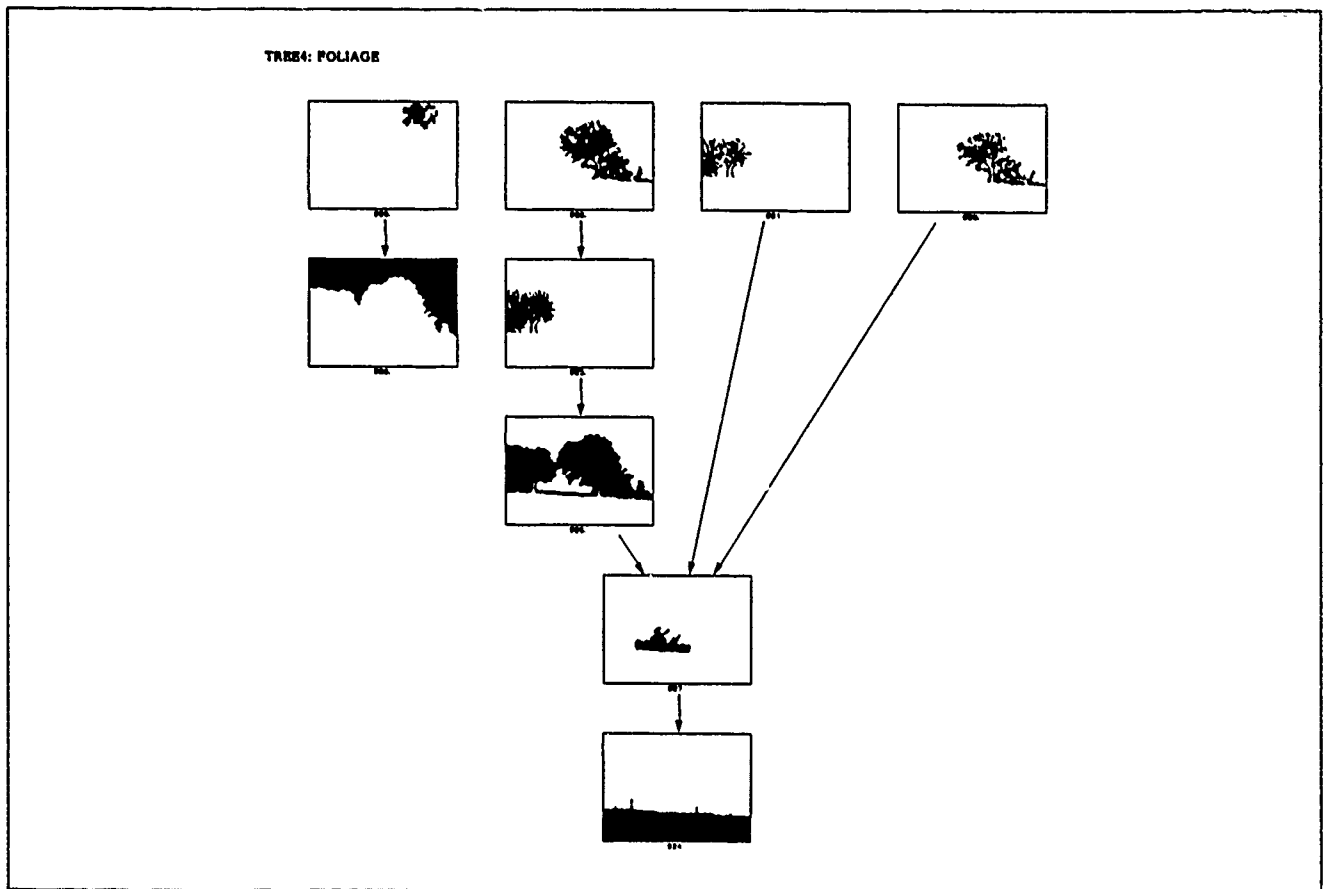


Figure 3: Partial order of candidates for foliage

the world. Without range data, Condor uses the image location of detected objects along with a digital terrain model stored in the CKS to constrain the possible locations of each object. This updated database is then used by Condor during analysis of subsequent images as context to aid interpretation.

5 Status and Future Plans

We are currently conducting an extensive series of experiments to test the validity of our ideas and to explore the limits of the implemented system. For purposes of this experimentation, we have concentrated on tailoring the context-set knowledge base to the task of recognizing natural objects in ground level images obtained from a two-square mile portion of the foothills behind the Stanford University campus. Our ultimate goal is for Condor to be able to understand the scene in any non-degenerate image acquired in this area. The natural objects occurring in this environment consist of trees, bushes, rocks, trails, and grass in addition to the sky and the ground.

Our intent is to develop a recognition capability that is on a par with that exhibited, say, by a rabbit, which inhabits the same environment. This requires the ability to recognize scenes under many conditions including

variations due to sun angle, weather, seasonal changes, normal plant growth, and discrete changes such as when a tree has fallen or a new trail has been blazed. A key issue in our research has been determining what contextual information should be recognized and stored to enable robust recognition under such a diversity of conditions.

We have taken over 100 photographs at the experimental site of which approximately 30 so far have been digitized and analyzed by Condor. This data includes monochrome as well as color imagery, and range data obtained from automatic compilation of binocular stereo pairs. A digital terrain model of the area and the information appearing on a USGS map provide initial context.

We are in the process of conducting a series of experiments to demonstrate the competence of the system and the value of contextual information during recognition.

Experiment 1: A single image is analyzed using only the initial context obtained from the map. Upon completion, Condor stores its recognition results in the CKS and reanalyzes the same image or a similar but different image of the same scene. In many cases, the recognition result is improved by using the newly acquired context. Repeated analysis of

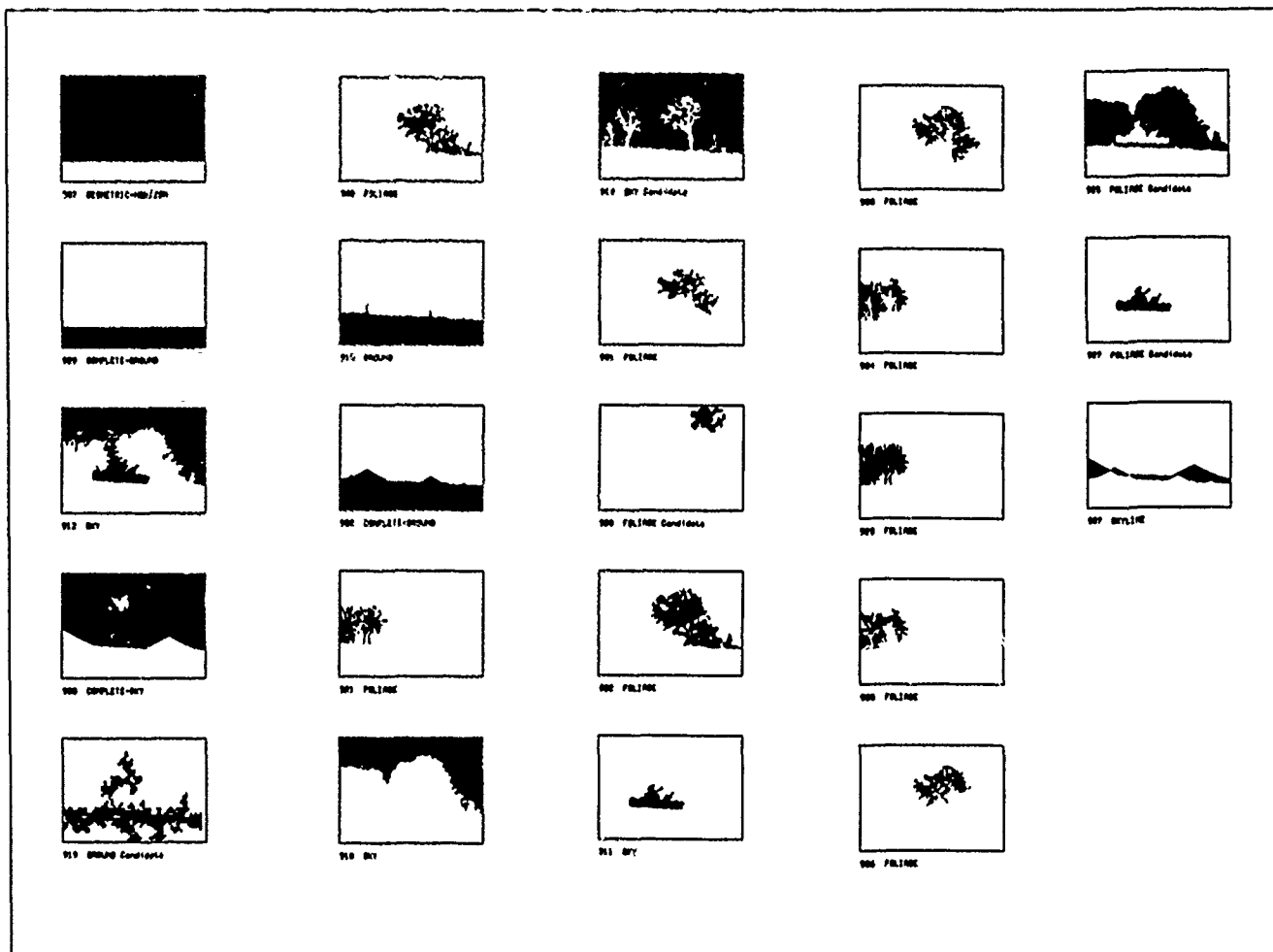


Figure 9: Sequence of candidates nominated for inclusion in Clique 1 (reading downward)

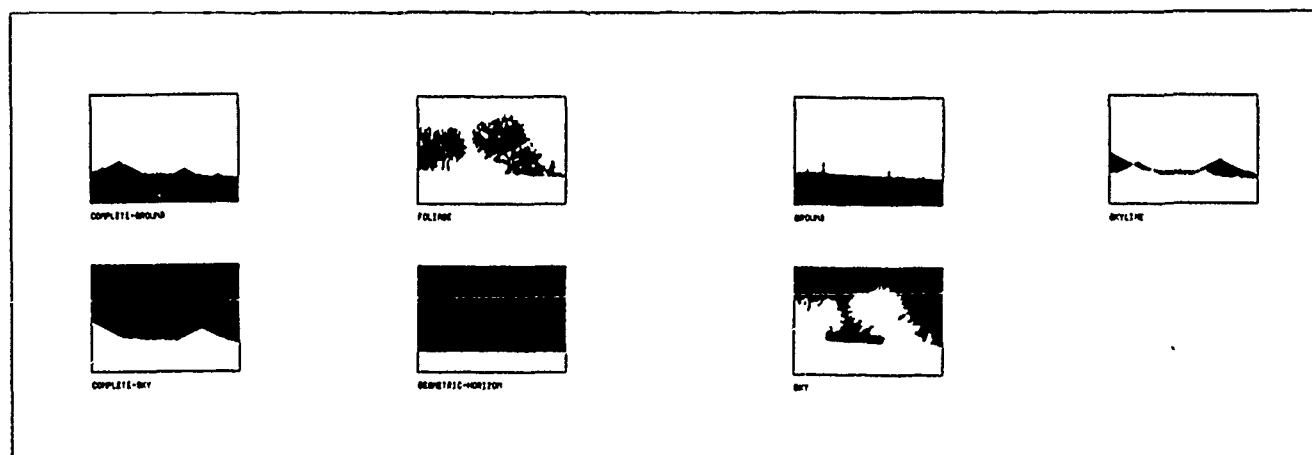


Figure 10: Composite labeling found by Clique 1

the image set leads to both faster recognition of known objects as well as detection of some previously unrecognized objects.

Experiment 2: A sequence of imagery collected during a simulated traverse of the terrain is analyzed by Condor. The results of processing each image are stored in the CKS and made available as context for analyzing subsequent images. The temporal continuity provided by the information in the CKS allows Condor to improve the results it would have obtained without this additional contextual information. Upon completion of the sequence, Condor has built a 3D model of the objects visible during the traverse, and has annotated each with information that aids its recognition.

Experiment 3: A collection of images from a restricted area but varying widely in viewpoint, scale, time-of-day, season, and sky conditions are analyzed by Condor. In most cases Condor obtains a consistent recognition result, demonstrating that the context-set knowledge base is insensitive to these types of change.

In the future, additional imagery will be digitized to more fully evaluate the approach. We expect to have well over 100 images analyzed by Condor, including some obtained only after further development of the knowledge base has been frozen. Based on our initial experimental results, and the unique architecture of our system, we are highly optimistic about the ability of Condor to overcome many of the limitations inherent in the traditional machine vision paradigms.

References

- [1] Barnard, Stephen T., "Stochastic Stereo Matching over Scale," *International Journal of Computer Vision*, 3(1), 1989.
- [2] Barrow, Harry G., and Tenenbaum, Jay M., "MSYS: A System for Reasoning about Scenes," Technical Note 121, Artificial Intelligence Center, SRI International, April 1976.
- [3] Bolles, R.C., Horaud, R., and Hannah, M.J., "3DPO: A 3D Part Orientation System," in *Proceedings 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, August 1983, pp. 1116-1120.
- [4] Brooks, Rodney A., "Model-Based 3-D Interpretations of 2-D Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 5, Number 2, March 1983, pp. 140-150.
- [5] Fischler, Martin A., and Strat, Thomas M., "Recognizing Trees, Bushes, Rocks and Rivers," *Proceedings of the AAAI Spring Symposium Series: Physical and Biological Approaches to Computational Vision*, Stanford University, March 1988, pp. 62-64.
- [6] Fischler, Martin A., and S. t., Thomas M., "Recognizing Objects in a Natural Environment: A Contextual Vision System," *Proceedings: DARPA Image Understanding Workshop*, Palo Alto, California, May 1989, pp 774 - 796.
- [7] Fua, Pascal, and Hanson, Andrew J., "Using Generic Geometric Models for Intelligent Shape Extraction," *Proceedings: DARPA Image Understanding Workshop*, Los Angeles, California, February 1987, pp. 227-233.
- [8] Hannah, M.J., "SRI's Baseline Stereo System," *Proceedings: DARPA Image Understanding Workshop*, Miami Beach, Florida, December, 1985, pp. 149 - 155.
- [9] Hanson, A.R., and Riseman, E.M., "VISIONS: A Computer System for Interpreting Scenes," in *Computer Vision Systems*, Academic Press, New York, 1978, pp. 303-333.
- [10] Hanson, A.J., and Quam, L., "Overview of the SRI Cartographic Modeling Environment," *Proceedings: DARPA Image Understanding Workshop*, Cambridge, Massachusetts, April 1988, pp. 576 - 582.
- [11] Huttenlocher, Daniel P., and Ullman, Shimon, "Recognizing Solid Objects by Alignment," *Proceedings: DARPA Image Understanding Workshop*, Cambridge, Massachusetts, April 1988, pp. 1114-1122.
- [12] Laws, Kenneth, L., "Integrated Split/Merge Image Segmentation," Technical Note 441, Artificial Intelligence Center, SRI International, July 1988.
- [13] Smith, Grahame B., and Strat, Thomas M., "Information Management in a Sensor-Based Autonomous System," *Proceedings: DARPA Image Understanding Workshop*, Los Angeles, California, February, 1987, pp. 170-177.
- [14] Strat, Thomas M. and Smith, Grahame, B., "A Knowledge-Based Information Manager for Autonomous Vehicles," Chapter 1 in *Image Understanding in Unstructured Environments*, edited by Su-shing Chen, World Scientific Publishing Co, 1988, pp 1-39.
- [15] Strat, Thomas M. and Fischler, Martin A., "Context-Based Vision: Recognition of Natural Scenes," *Proceedings of the 23rd Asilomar Conference on Signals, Systems, and Man*, October, 1989, pp 532-536.

Region Grouping Using the Minimum-Description-Length Principle

Yvan G. Leclerc

Artificial Intelligence Center
SRI International
333 Ravenswood Ave., Menlo Park, CA 94025

Abstract

A new algorithm for region grouping is presented. This algorithm is based on an old idea—simplicity of description—but uses a relatively new mathematical formalism: the Minimum-Description-Length Principle. Also presented is a brief outline of how this principle can be applied to vision in a hierarchical fashion. Previous research in implementing the first layer of this proposed hierarchy (image segmentation) is briefly reviewed.

1 Introduction

A basic task for any visual system is to infer properties of the outside world given a stream of incoming images. The primary difficulty is that there are always an infinite number of combinations of world properties (e.g., surface shape, albedo, and illumination) that can produce the same image. Thus, one cannot simply deduce these properties from an image. Instead, one must choose a single combination according to some guiding principle.

We hypothesize that an important guiding principle in vision is simplicity of description, or, more formally, the Minimum-Description-Length (MDL) Principle (Chaitin 1966, Minsky 1962, Rissanen 1978, Solomonoff 1964). According to this principle, prior information about the world and image sensor is incorporated in the language used to describe the world and sensor, and the inference process is to find the simplest (i.e., shortest) description in the language that exactly reproduces the given images. The basic motivation behind this inference process is the hypothesis that, if one can find a language that provides an efficient description of a large number of observations (images), then the simplest descriptions in that language tell us something about the causes of the observations.

The general notion of simplicity of description in vision has been very much a part of the field since the Gestalt psychologists' exposition of the principle of *Prägnanz* (Koffka 1935). Hochberg (1981) and others distinguish between two different forms of *Prägnanz*: *simplicity* and *likelihood*. To quote Hochberg (1981, p. 263), the likelihood principle states that "we perceive whatever object

or scene would, under normal conditions, most likely fit the sensory pattern we receive," whereas the simplicity principle states that "we perceive whatever pattern or object would most simply or economically fit the sensory pattern." The debate between these two forms of *Prägnanz* can be resolved by using the MDL principle because the simplicity and likelihood principles can be made equivalent (Chaitin 1975).

Since the Gestalt psychologists, many researchers have defined visual perception in terms of simplicity of description (Attneave 1954, Hochberg and McAlister 1953, Leeuwenberg 1971). Almost exclusively, however, this work has dealt with the coding of schematic line drawings of two- and three-dimensional objects, rather than images of the objects. Although the kinds of encoding schemes they advocate are probably important, it seems that the most important applications of the principle of simplicity will be in going from the image to such schematic representations.

We begin with a brief exposition of how the MDL principle could be applied to the general vision problem. Following this, we briefly describe our previous research in applying the MDL principle to one of the first stages in image analysis (image and boundary segmentation). Finally, we describe in more detail the specific application of the principle that is the main focus of this paper: region grouping.

For a more complete presentation of the ideas, algorithms, and results described herein, see (Leclerc 1988, Leclerc 1989a, Leclerc 1989b, Leclerc 1989c, Leclerc 1990). For other applications of the minimal-length encoding approach to vision, see (Keeler 1990, Pednault 1989, Pentland 1990, Pentland and Darrell 1990, Sander-son and Foster 1990, Smith and Wolf 1984).

2 Applying the MDL Principle to Vision

One way of formalizing the MDL principle for the vision problem is as follows. Define a model for surface shape such that one can define a measure of the com-

plexity of any given surface. Typically, the complexity measure will be proportional to the number of parameters required to specify the surface. For example, one could use three-dimensional polynomials to describe a surface. In this case, a planar surface (requiring four coefficients) would be simpler than a wrinkled surface such as the skin of an orange (requiring dozens of coefficients). (Note that the complexity of a surface depends entirely on the model we choose. If we had chosen fractals, say, instead of polynomials, the complexity of the planar surface would be quite different.) Similarly, define a model for the distribution of albedo on a surface and for the distribution of light sources. Of the many combinations of shape, albedo, and lighting that can produce a given image, the principle of simplicity of description is to choose that combination for which the sum of the complexities of each of the components is least. (This presupposes that the three components are independent, which is usually a good approximation.) In short, one chooses the simplest way of describing a given image given the set of models available to us.

The above principle cannot, in general, be implemented exactly because it involves an exhaustive search through the set of all possible combinations of shape, albedo, and illumination. The combinatorial nature of the problem necessarily leads to approximate solution techniques that typically find the simplest description in a limited variety of circumstances only.

Thus, to implement the simplicity principle, one must do two things: one must define a set of models, and one must define a set of approximate solution techniques. As is traditional in the computer vision field, we hypothesize that the search for the simplest description takes place in a hierarchical, layered fashion. Furthermore, we hypothesize that each layer can itself be viewed as a search for the simplest description, but using a more constrained set of models.

This approach has led to a high-performance image segmentation and reconstruction algorithm, which we view as the first (or, at least, one of the first) layer in the hypothesized descriptive hierarchy (Leclerc 1989a, Leclerc 1989c). The algorithm has been implemented on a massively parallel architecture (the Connection MachineTM). It has been tested on a large number of synthetic and natural images, and its performance compared against standard techniques such as the Canny edge operator.

From these tests we conclude that, without adjusting any parameters, the algorithm is capable of finding region boundaries whose contrast-to-noise ratio is significantly less than one. High contrast boundaries are accurately located to within a pixel and are as smooth or ragged as those of the true underlying image, while low contrast ones are smooth. Also, because it is a global optimization technique, the algorithm can even close boundaries across gaps that have zero local con-

trast (what one might call subjective edges). Furthermore, the algorithm is adaptive in the sense that the variance of the noise need not be known a priori and can even be non-uniform across the image. It is also adaptive in the sense that it automatically determines the appropriate amount of smoothing required at every point in the image.

A brief description of this first layer, the approximate solution technique used, and some results illustrating the claims made above are presented in the following section.

3 The First Descriptive Layer

In the first descriptive layer, the image is decomposed into an underlying piecewise-polynomial image and spatially varying white noise. The underlying image is described as a set of regions covering the entire image; the boundary of each region is described using a chain code, and the interior intensity variation is described by a polynomial. The complexity of the underlying image is the sum of the complexities of each region, where the complexity of a region is a linear combination of the length of the region boundary and the number of non-zero coefficients of the polynomial. In this implementation, neither the number of regions, nor the length and shape of the region boundaries, nor the number of coefficients in each region is known ahead of time. The difference between this underlying image and the given image is modeled as white noise whose variance is unknown and independent from one region to the next. The theoretical complexity of the noise for each region is roughly $n \log \sigma$, as derived by standard information theoretic means. By construction, the underlying image and noise completely describe the original image.

The task of this first layer is thus to find the underlying image for which the overall complexity, the complexity of the underlying image plus the complexity of the noise, is least. Since the noise is simply the difference between the underlying image and the given image, one need only search over the set of all possible underlying images. However, this set is exponential in the number of pixels in the image, so that the simplest description cannot generally be found in less than exponential time.

This search problem can be recast as an optimization problem in which the objective function is an approximation to the overall complexity. This is done by using a finite-element grid to represent the underlying image. Each element of this grid represents a polynomial within a unit square. By specifying the coefficients of the polynomials for each element, we directly specify the underlying image, and implicitly specify the set of regions. That is, two adjacent elements are defined to be within the same region if the two sets of coefficients, when expressed in a common coordinate system, are identical.

With this representation, the objective function (overall complexity) can be written as the sum of spatially lo-

cal terms that involve only the polynomial coefficients of an element, those of its four neighbors, and the pixel of the given image. Finding the vector of coefficients that minimizes this objective function is difficult because the objective function has exponentially many local minima, so that standard optimization techniques cannot be used.

Instead, we devised an approximate solution technique based on a general approach called a "continuation method." (Dahlquist and Björck 1974, Terzopoulos 1986, Blake and Zisserman 1987). In this approach, the original objective function is embedded in a family of functions parameterized by s . This embedding has the property that for sufficiently large s , the function has a single local minimum that is readily found, while for s arbitrarily close to zero, the function is arbitrarily close to the original objective function. The continuation method starts at the local minimum found for a large value of s , and tracks the minimum as s is progressively decreased. Specifically, the local minimum found at a given value of s is used as the starting point for a kind of steepest descent algorithm applied to the embedded objective function for a smaller fixed value of s . This is repeated until s is sufficiently close to zero.

The advantage of having written the original objective function as the sum of spatially local terms is that the resulting descent algorithm is entirely parallelizable. That is, at each iteration of the descent algorithm, the change in the coefficients of the polynomial for a given grid element can be computed using only the coefficients of the adjacent elements and the corresponding image pixel value. This kind of parallel and iterative algorithm is perfectly suited to massively parallel architectures, such as the Connection MachineTM that was used for the current implementation.

3.1 An example

An example of the result of this algorithm is illustrated in Fig. 1. Fig. 1a is the original image, which consists of a set of white scalloped disks on a black background, embedded in additive white noise. The variance of the noise increases linearly from left to right. Before the addition of the noise, all of the disks had exactly the same shape. Fig. 1b is the underlying piecewise-constant image recovered by the algorithm, and Fig. 1c is the residual image, representing the difference between the original image and the recovered piecewise-constant image. Note that the first column of disks, where the *local* signal-to-noise ratio is quite high, have been recovered exactly—all of the recovered disks have the same shape. In the next column, the shapes are quite similar, but not exactly so. As the local signal-to-noise ratio decreases, the shapes of the disks become more dissimilar, but also smoother, as is clear in the final column.

From this example we see that when the signal-to-noise ratio is sufficiently high, every detail of the dis-

continuities in the underlying image is preserved. As the image is degraded, the precise details are sometimes lost, but the procedure continues to find the distinct regions in the underlying image. (Eventually, of course, the regions are lost entirely. The precise point at which this occurs depends on the total size and shape of the regions, and, to a lesser extent, on the vagaries of the noise.)

For other examples using the more complex piecewise-polynomial model, see (Leclerc 1989a), and the examples discussed in the following section.

4 The Next Descriptive Layer—Grouping

For a large class of scenes, each region recovered by the above segmentation algorithm will correspond to a single *part* of some surface in the scene. However, because of albedo changes, shadows, creases, and partial occlusions, a single surface will be broken up into many regions. The next simplification stage is thus to group together regions that belong to a single surface. In this section, we explore in detail one possible basis for grouping regions that originally belonged to a single surface in the scene.

Consider Fig. 2(a). It was constructed as a middle-grey annulus against a light background, occluded by a darker surface with regularly spaced holes, all of which is embedded in white noise. This construction corresponds to one possible percept of this image. The other percept that I am aware of is that of a regularly spaced grid of light disks on a dark background, partially occluded by a translucent grey annulus. In either case, one immediately groups the middle-grey regions (composed of disks and parts of disks) into one "object," and groups the light regions into a second object.

On what basis can we perform this grouping of regions? One possible basis is the "good continuation" of segments of the region boundaries (Wertheimer 1955, Lowe 1985, Zucker 1985). Within the MDL paradigm, we interpret good continuation to mean that it is simpler to describe these segments as parts of a single curve than as independent curves. For example, Fig. 2(c) shows the boundaries of the recovered regions. From this image, it seems quite plausible that one could describe the segments of the outer perimeter of the annulus more efficiently by using a single curve (i.e., a circle) than as independent curves, thereby recovering the outer perimeter of the annulus from the region boundaries alone. But it seems rather implausible that the inner perimeter could be so recovered, given that there are only four unoccluded segments of the inner perimeter. We shall not further explore good continuation of region boundaries in this paper.

A second possible basis is the "good continuation" of the intensity variation within the regions. In this case,

we interpret good continuation to mean that the intensity variation within a group of regions is simpler to describe using a single polynomial model than with the independent polynomials (one per region) originally recovered by the segmentation algorithm. For a piecewise-constant image such as Fig. 2(a), this basically means that regions with roughly the same intensity should be grouped together. Figs. 2(d,e,f) show the three resulting groups of regions found by the grouping algorithm (the algorithm will be described shortly). The first group of regions correspond to the light background regions, the second group to the middle-grey regions of the annulus, and the final group (a single region with many holes, shown in black), corresponds to the occluding grid.

Fig. 3 illustrates the results of the segmentation and grouping algorithms for a piecewise-first-order image. In this case, a different polynomial was used for the regions within the inner perimeter of the annulus, resulting in four groups of regions.

As a final illustration, Fig. 4(a) is an image of tree branches against the sky. Even in such a low-resolution image one immediately perceives the occlusion of the sky by the branches, and groups the sky regions together as one "object" in the scene. Furthermore, because of the nature of the occluding objects (tree branches), it seems quite implausible that any kind of good continuation of the boundaries could lead to the correct grouping.

Note that the sky has a strong gradient, being brighter in the lower right corner of the image, and darker in the upper left. Thus, a piecewise-constant model (such as simple thresholding) is insufficient. A piecewise-first-order model was used. The recovered piecewise-first-order image is shown in Fig. 4(b), and Fig. 4(c) shows one of the groups of regions recovered by the grouping algorithm. Note that most of the larger occluded sky regions have been correctly grouped together.

4.1 The Grouping Algorithm

Informally, a group of regions is defined as a set of regions for which the intensities within all of the regions of the group are modeled as the sum of a single polynomial and white noise. The cost of encoding the intensities in a group is thus the cost of encoding the p non-zero coefficients of the polynomial plus the cost of encoding the residuals (the point-by-point differences between the polynomial and the original image intensities).

A complete grouping of an image is defined as an exhaustive set of mutually exclusive groups. That is, each region in an image belongs to exactly one group, and each group must contain at least one region. An optimal grouping is defined as a complete grouping for which the cost of encoding the intensities (which is the sum of the costs for each group) is least among all possible groupings. If there are $|R|$ regions produced by the segmentation algorithm, then the number of possible groups

quickly grows beyond $2^{|R|}$, making the search for the optimal group computationally infeasible. Although this number is much smaller than the number of possible segmentations, it is still too large to search through explicitly. Thus, as with the segmentation problem, we need to use an approximate solution technique.

We now define a continuation method that is quite similar to the one used by the segmentation algorithm. However, rather than having a node for each pixel in the image, we now have a node for each region. The node is a vector representing the coefficients of the polynomial for the entire region. For convenience, a local coordinate system is defined for each region, with the origin at the centroid of the region. Rather than having each node linked in a graph to its four (or eight) spatially nearest neighbors as in the segmentation algorithm, each node is now linked to the k nearest regions according to a metric which includes both the spatial distance between the regions and the difference between the coefficients of the independent polynomials. The purpose of the continuation method is to determine which links to maintain. When the continuation method is complete, the groups are defined to be the connected subgraphs.

In principle, each region should initially be linked to all other regions, but, to reduce the computational burden, only the k nearest regions (as defined above) are used. Of course, if two regions in the optimal grouping are not a part of a connected subgraph of the initial graph, then the continuation method will fail to find the optimal grouping. Thus, to minimize this possibility, k must be chosen sufficiently large so that any errors induced by the heuristic use of the pair-wise metric will not disconnect regions that belong to the optimal grouping. In practice, $k = 16$ seems to be sufficient.

Formally, let z_i denote the intensity of the original image at the i^{th} pixel, located at coordinate (x_i, y_i) (a single index is used to simplify the notation). Let R_r denote the set of pixels in the r^{th} region recovered by the segmentation algorithm. Let G_g denote a group of regions. The cost of encoding the intensities in a group is the sum of (1) the cost of encoding the p non-zero coefficients of the polynomial and (2) the cost of encoding the residuals. The cost of encoding the region boundaries, required for the segmentation algorithm, is ignored here because the boundaries remain fixed.

As derived by Rissanen (1983), the cost of encoding the p non-zero coefficients of the polynomial is of order¹

$$\frac{p}{2} \log n, \quad (1)$$

where n is the total number of pixels in all of the regions in the group. Because Eq. 1 is accurate only when n is sufficiently large, this causes some difficulties in directly applying this formula for segmentation and grouping. In particular, when n is small, this formula significantly

¹ Unless otherwise noted, logarithms are base 2.

underestimates the encoding cost. In practice, adding a constant to the formula is a better approximation.

The cost of encoding the residuals is proportional to (Leclerc 1989a)

$$\sum_{r:R_r \in G_g} \sum_{i \in R_r} \left(\frac{z_i - u_r(i)}{\sigma_g} \right)^2,$$

where $u_r(i)$ is the value of the least-squares polynomial evaluated at the i^{th} pixel,² and σ_g is the measured variance in the group.

As with the segmentation algorithm, the cost of encoding the coefficients must be approximated locally because n , the number of pixels in the group, is unknown until the grouping is complete. We use the following approximation, in which we define the cost of breaking a link between two regions as the difference in encoding costs between using a separate polynomial for each region versus a single polynomial for both regions. This cost is well-defined because the area of each region is known from the start. For simplicity of exposition, we derive the costs and continuation method for the piecewise-constant case when the variance of the noise is fixed. In this case, $p = 1$, and the polynomial coefficient corresponds to the mean of the regions. The more general case can be derived in the same fashion as the segmentation algorithm (Leclerc 1989a).

If we were to encode two regions, R_r and $R_{r'}$ separately, then we would need

$$\frac{1}{2} \log |R_r| + c$$

bits to encode the mean for region R_r , and

$$\frac{1}{2} \log |R_{r'}| + c$$

bits to encode the mean for region $R_{r'}$. If we now encode the two regions together, we only need

$$\frac{1}{2} \log (|R_r| + |R_{r'}|) + c$$

bits for the mean of the combined regions. Thus, we define the cost of keeping two regions separate as the difference between these costs:

$$b = \frac{1}{2} [\log |R_r| + \log |R_{r'}| - \log (|R_r| + |R_{r'}|) + 2c].$$

We choose c by demanding that this expression be positive in the extreme case when both regions are one pixel in area. Thus, we choose $2c = \log 2$, and get

$$b = \frac{1}{2} \log \left(\frac{2|R_r||R_{r'}|}{|R_r| + |R_{r'}|} \right). \quad (2)$$

²For a fixed p , the least-squares polynomial minimizes the encoding cost for a white-noise model (Leclerc 1989c). In general, p is a variable that must be determined as part of the optimization process, as it was for the segmentation algorithm. For the purposes of this paper, we assume that p is fixed.

This choice of c has the interesting property that it makes this cost equal to the cost of introducing a discontinuity in the segmentation algorithm. Thus, the segmentation algorithm can be viewed as a special case of grouping where all the regions are one pixel in area, and the regions are initially linked in a regular fashion.

Using the above local approximation to the cost, we define the following objective function for a given set of neighborhood relations (graph) N :

$$\begin{aligned} L(u, N) &= a \sum_r \sum_{i \in R_r} \left(\frac{z_i - u_r(i)}{\sigma} \right)^2 \\ &\quad + \frac{b}{2} \sum_r \sum_{r' \in N_r} (1 - \delta(u_r - u_{r'})) \\ &= a \sum_r |R_r| \left(\frac{m_r - u_r}{\sigma} \right)^2 \\ &\quad + \frac{b}{2} \sum_r \sum_{r' \in N_r} (1 - \delta(u_r - u_{r'})) \quad (3) \end{aligned}$$

where N_r is the set of regions linked to region R_r , b is from Eq. 2, and m_r is the mean of z_i over the region R_r . (Note that b is divided by 2 because links are counted twice.)

The first term in Eq. 3 is the cost of the encoding the residuals, and the second term is the cost of breaking a link between neighboring pairs of regions. (A link is defined to be broken whenever $u_r \neq u_{r'}$.)

Note that when the graph is fixed, the objective function is a simple quadratic with the global minimum at

$$u_r = \frac{\sum_{r' \in N_r} |R_{r'}| m_{r'}}{\sum_{r' \in N_r} |R_{r'}|}.$$

That is, u_r is the mean of the image intensities over the group of regions that R_r belongs to, as expected.

The purpose of the continuation method is to determine the correct neighborhood relations by eliminating links from the initial graph. That is, the continuation method must determine the values of the u_r such that $u_r = u_{r'}$ when regions R_r and $R_{r'}$ belong to the same group for the optimal grouping, and u_r equals the mean of the intensities within the group that it belongs to.

The continuation method embeds the original objective function $L(u, N)$ in a family of functions $L(u, N, s)$ such that $L(u, N, 0) = L(u, N)$. The specific embedding used here, as in the segmentation algorithm, replaces $\delta(u_r - u_{r'})$ with an exponential,

$$\delta(u_r - u_{r'}) \rightarrow e_{r,r'}(u, s) \equiv \exp \left(-\frac{(u_r - u_{r'})^2}{(s\sigma)^2} \right),$$

so that

$$\begin{aligned} L(u, N, s) &= a \sum_r |R_r| \left(\frac{m_r - u_r}{\sigma} \right)^2 \\ &\quad + \frac{b}{2} \sum_r \sum_{r' \in N_r} (1 - e_{r,r'}(u, s)). \quad (4) \end{aligned}$$

We begin with a large value of s , for which Eq. 4 is a quadratic with a single global minimum at $u_r = m_r$. This local minimum is tracked as a decreasing function of s , and the local minimum arrived at for a sufficiently small value of s is the solution.

More specifically, we start with a large value s^0 , and apply the following iterate,

$$u_r^{t+1} = \frac{m_r + \frac{b}{a(s^{t+1})^2} \sum_{r' \in N_r} e_{r,r'}(u^{t+1}) u_{r'}^t}{1 + \frac{b}{a(s^{t+1})^2} \sum_{r' \in N_r} e_{r,r'}(u^{t+1})}, \quad (5)$$

with $s^{t+1} = s^t$ until $|u_r^{t+1} - u_r^t| < \epsilon$ (see (Leclerc 1989a) for the derivation of this Gauss-Jordan iterate). When the iterate has stabilized, u^{t+1} is at a local minimum of $L(u, N, s^{t+1})$. Then, s is reduced by letting $s^{t+1} = \alpha s^t$ for some positive α less than 1. All of this is repeated until s^{t+1} becomes sufficiently close to 0.

5 Summary and Conclusions

The significance of the MDL approach is that it can deal with all aspects of the visual interpretation problem (delineation, partitioning, recognition, etc.) in a uniform and principled manner. It appears to be computationally feasible with current computing technology and has been demonstrated to be effective in situations where conventional techniques fail.

We have presented one application of the MDL principle to vision. This application was presented as one possible layer in a hierarchy, where each layer would produce an increasingly more compact representation of the previous layer (the first layer being the image itself). The second layer of this hierarchy is image segmentation, which was briefly reviewed here. The third layer is region grouping, which was described in detail in this paper.

In conclusion, the simplest description principle is a new paradigm in computer vision that has produced impressive results with respect to some of the problems of early vision, and it is currently being extended to the later stages of visual interpretation.

Acknowledgments. The work reported here was partially supported by the Defense Advanced Research Projects Agency under contract MDA903-86-C-0084, DACA76-85-C-0004, and 89F737300. Use of the Connection Machine(TM) was supported by the Defense Advanced Research Agency.

References

Attneave, F. (1954). Some informational aspects of visual perception. *Psychological Review*, 61, 183-193.

Blake, A. and Zisserman, A. (1987). *Visual Reconstruction*. MIT Press, Cambridge, Massachusetts.

Chaitin, G. J. (1966). On the length of programs for computing finite binary sequences. *Journal of the ACM*, 19, 547-569.

Chaitin, G. J. (1975). A theory of program size formally identical to information theory. *Journal of the ACM*, 22, 329-340.

Dahlquist, G. and Björck, A. (1974). *Numerical Methods*. Prentice Hall, Englewood Cliffs, New Jersey.

Hochberg, J. (1981). Levels of perceptual organization. In *Perceptual Organization*, pages 255-278, Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Hochberg, J. and McAlister, E. (1953). A quantitative approach to figure 'goodness'. *Journal of Experimental Psychology*, 46, 361-364.

Keeler, K. (1990). Minimal-length encoding of planar subdivision topologies with application to image segmentation. In *Working Notes of the AAAI Spring Symposium on the Theory and Application of Minimal-Length Encoding*, pages 95-99, AAAI, Stanford University.

Koffka, K. (1935). *Principles of Gestalt Psychology*. Harcourt, Brace and World, New York, New York.

Leclerc, Y. G. (1988). Constructing simple stable descriptions for image partitioning. In *Proceedings of the 1988 DARPA Image Understanding Workshop*, pages 365-382, Cambridge, Massachusetts.

Leclerc, Y. G. (1989a). Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3(1), 73-102.

Leclerc, Y. G. (1989b). Image and boundary segmentation via minimal-length encoding on the connection machine. In *Proceedings of the 1989 DARPA Image Understanding Workshop*, Palo Alto, California.

Leclerc, Y. G. (1989c). *The Local Structure of Image Intensity Discontinuities*. PhD thesis, McGill University, Montréal, Québec, Canada.

Leclerc, Y. G. (1990). Simplicity of description as the basis for visual interpretation. In *Working Notes of the AAAI Spring Symposium on The Theory and Application of Minimal-Length Encoding*, American Association for Artificial Intelligence, Stanford University, Palo Alto, California.

Leeuwenberg, E. L. J. (1971). A perceptual coding language for visual and auditory patterns. *American Journal of Psychology*, 84, 307-349.

- Lowe, D. (1985). *Perceptual Organization and Visual Recognition*. Kluwer Academic, Boston, Massachusetts.
- Minsky, M. L. (1962). Problems of formulation for artificial intelligence. In Bellman, R. E., editor, *Mathematical Problems in the Biological Sciences, Proceedings of Symposia in Applied Mathematics XIV*, page 35ff, American Mathematical Society, Providence, Rhode Island.
- Pednault, E. P. D. (1989). Some experiments in applying inductive inference principles to surface reconstruction. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*.
- Pentland, A. (1990). Automatic extraction of deformable part models. *International Journal of Computer Vision*, 4(2), 107-126.
- Pentland, A. and Darrell, T. (1990). Part segmentation for object recognition. In *Working Notes of the AAAI Spring Symposium on the Theory and Application of Minimal-Length Encoding*, pages 105-109, AAAI, Stanford University.
- Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, 14, 465-471.
- Rissanen, J. (1983). A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11, 416-431.
- Sanderson, A. C. and Foster, N. J. (1990). Attributed image matching using a minimal length criterion. In *Working Notes of the AAAI Spring Symposium on the Theory and Application of Minimal-Length Encoding*, pages 110-114, AAAI, Stanford University.
- Sinith, G. B. and Wolf, H. C. (1984). *Image-to-Image Correspondence: Linear-Structure Matching*. Technical Note 331, SRI International.
- Solomonoff, R. J. (1964). A formal theory of inductive inference. Parts I and II. *Information and Control*, 7, 1-22, 224-254.
- Terzopoulos, D. (1986). Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8, 413-424.
- Wertheimer, M. (1955). Law of organization in perceptual forms. In Ellis, W. D., editor, *A Source Book of Gestalt Psychology*, Routledge and Kegan Paul, London, England. First published in German, 1923.
- Zucker, S. W. (1985). *The Diversity of Perceptual Grouping*. Technical Report TR-85-1R, Computer Vision and Robotics Laboratory, McGill University, Montréal, Québec, Canada.

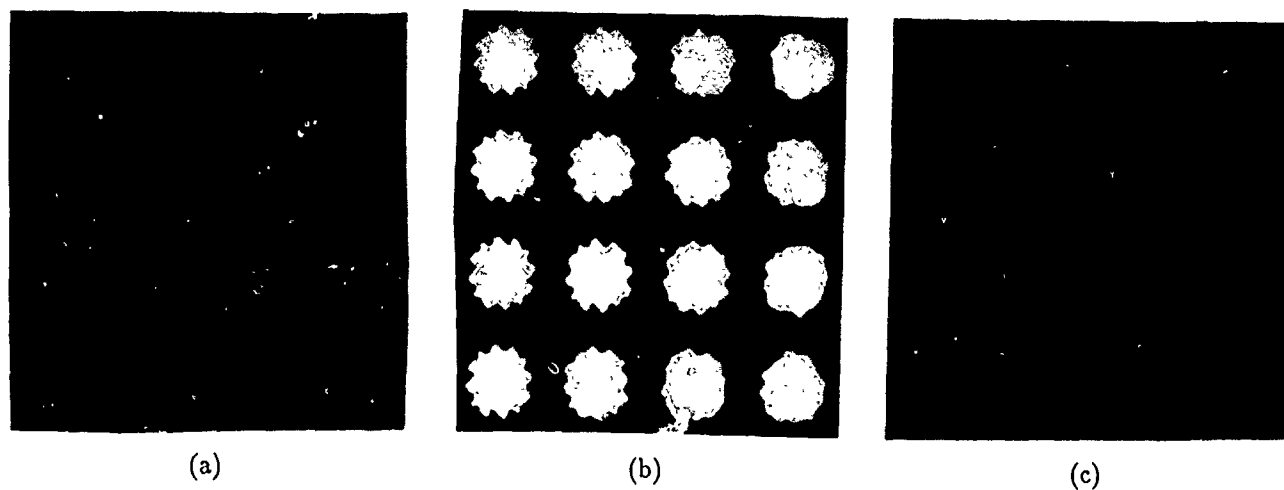


Figure 1. An illustration of the segmentation algorithm. (a) The input image. (b) The recovered piecewise-constant image. Note that the boundaries of the recovered regions are identical in the first column, where the local signal-to noise ratio (SNR) is high. As the local SNR decreases because the variance of the noise increases from left to right, the stability of the boundaries decreases, and the boundaries become smoother. (c) The residual image, defined as the difference between the original image (a) and the underlying image (b). Note that there is no perceptible structure remaining in the residuals, which indicates that the recovered image is correct.

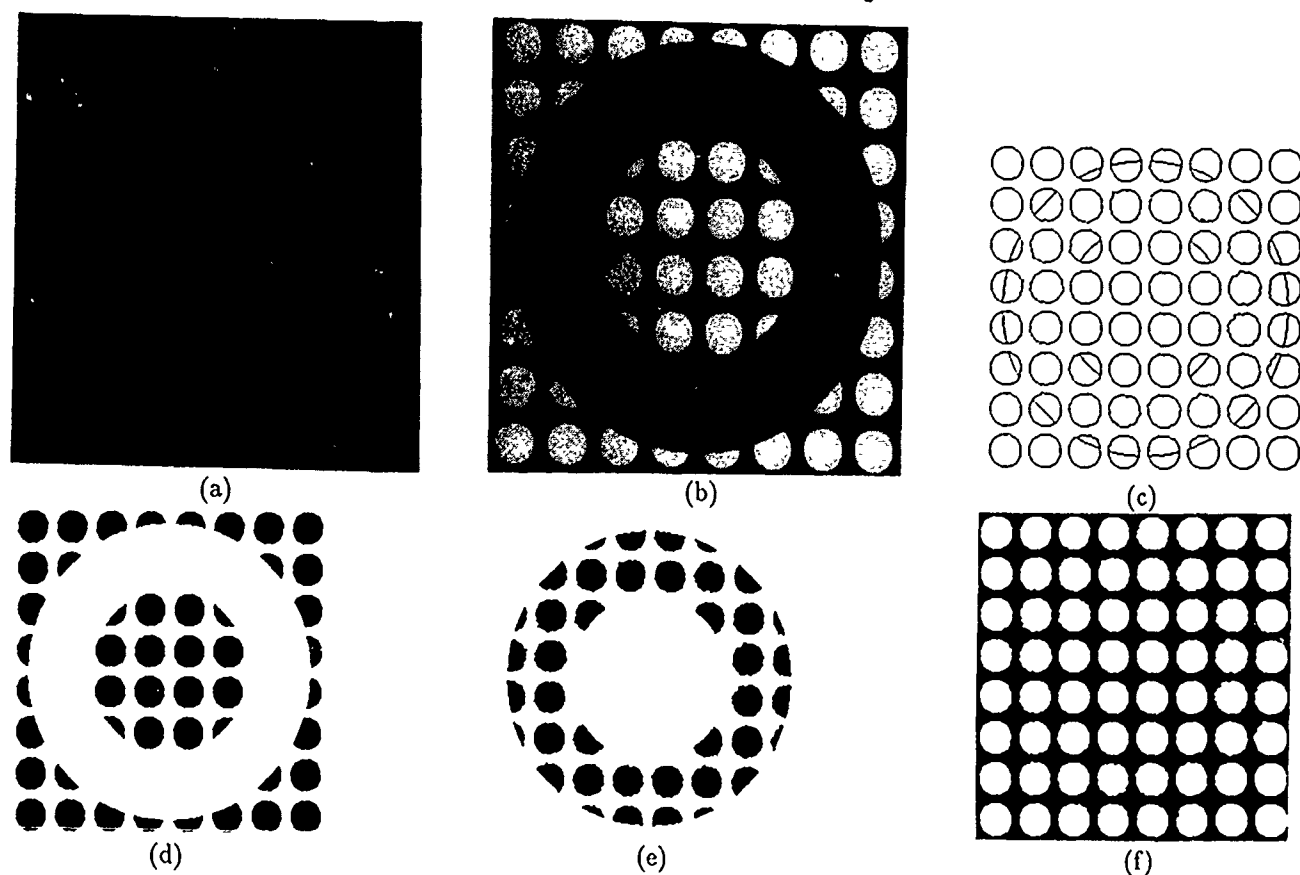


Figure 2. An illustration of the segmentation and grouping algorithms on a piecewise-constant image (a) The input image. (b) The underlying image recovered by the segmentation algorithm. (c) The boundaries of the regions recovered by the segmentation algorithm prior to grouping. (d) Shown in black is one group of regions that can be more compactly described using a single intensity model, as found by the grouping algorithm (e) A second group (f) The last group, composed of the single region (shown in black) with many holes.

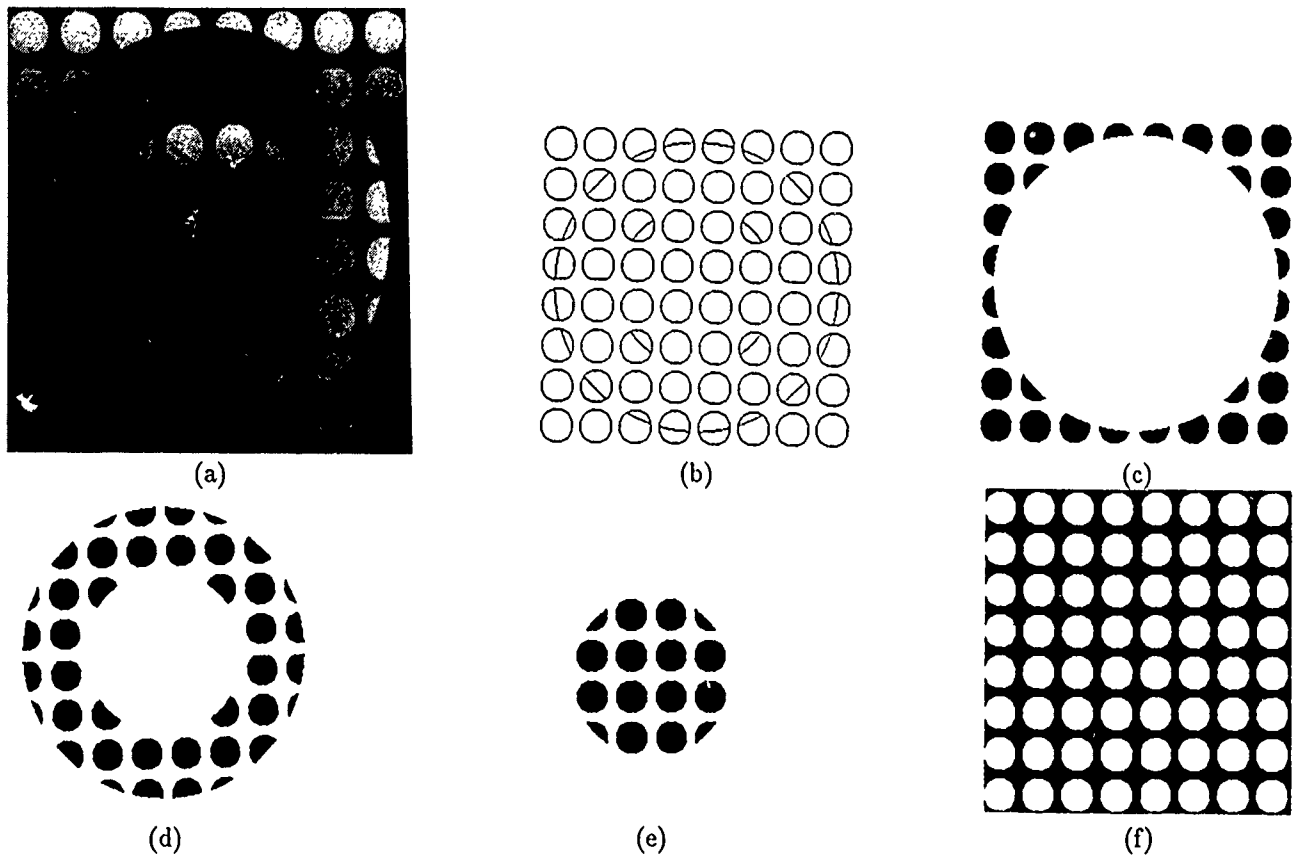


Figure 3: An illustration of the segmentation and grouping algorithms on a piecewise-first-order image. (a) The input image. (b) The boundaries of the underlying image recovered by the segmentation algorithm. (c,d,e,f) The groups of regions that can be more compactly described using a single intensity model, as found by the grouping algorithm.

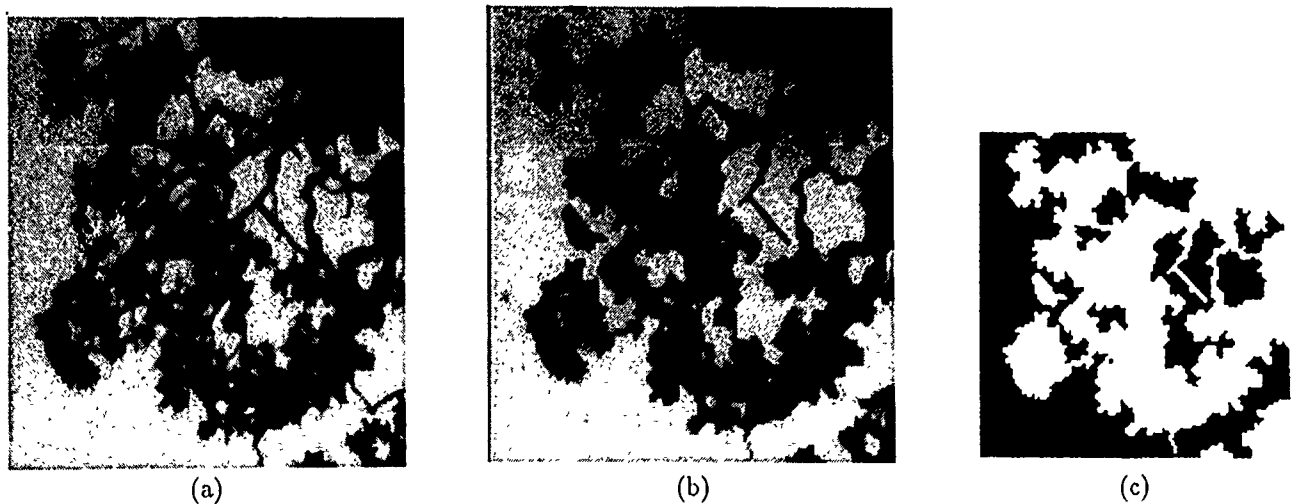


Figure 4: An illustration of the segmentation and grouping algorithms on a real image of tree branches against the sky. Note that there is a sufficient gradient in the sky that simple thresholding or a piecewise-constant model are inadequate. A piecewise-first-order model was used here. (a) The input image. (b) The underlying image recovered by the segmentation algorithm. (c) One of the groups of regions, found by the grouping algorithm. Note that this group contains most of the sky regions.

How to decide from the first view where to look next

Jasna Maver* and Ruzena Bajcsy*

GRASP Laboratory

Computer and Information Science Department

University of Pennsylvania

Philadelphia, PA 19104

Abstract

The task we want to achieve is the description of a random arrangement of unknown objects in a scene. First, a complete spatial map of the scene has to be acquired. To resolve the ambiguities that are caused by occlusions in range images, we need to take sensor measurements from several different views. We have limited ourselves to the images obtained by a laser scanning system which possesses certain features — occluded regions which are easily detected and can be used in designing an efficient algorithm. We develop a strategy to determine the sequence of different views using the information in a narrow zone around the occluded regions. Occluded regions are approximated by polygons. Based on the height information of the border of the occluded regions and geometry of the edges of the polygonal approximation, the next views are determined.

1 Introduction

The task we want to achieve is the description of a random arrangement of unknown objects in the scene. Multiple views [Bajcsy, 1988] are required because of geometrical complexity of the opaque objects in the scene. The problem of acquisition of multiple views, "where to look next", can be decomposed into several sub-problems. Developing the strategy for choosing the next viewing directions is perhaps the most challenging one. Another problem is: how to merge information from different views. This problem is closely related to object representation. We can merge the 3-D information obtained from each view into a common coordinate frame. When the complete spatial map is obtained it can be modeled by volumetric primitives such as superquadrics [Solina and Bajcsy, 1990]. Another approach is to find some topological features (vertices, edges, faces, etc.) which represent the basic elements for building the model of a scene. Based on incomplete information of different

views the structure of relations between these topological features may be constructed. An important problem which arises in merging 3-D information from different views is how to deal with the errors which appear in measurements and transformation parameters of applied transformations [Krotkov and Kories, 1988].

We limit ourselves to planning the next view. Some work has already been done on the planning of visual sensors [Sakane *et al.*, 1987a; Sakane *et al.*, 1987b]. Here the plan generation is done on the bases of geometrical and physical knowledge of the environment and models. This approach is model-based. N. Ahuja and J. Veenstra [1989] construct a 3-D model from the orthographic projections of an object onto a plane perpendicular to a viewing direction. Thirteen views must be selected from any subset of directions corresponding to the three "face" views, six "edge" views, and four "corner" views of an upright cube.

In our work we use range data. The first range image can be obtained from any direction in the scanning plane of our sensor system. All the next views are defined from the incomplete information of the previous views, and we also take into account the properties of our sensor system. We made experiments on real data.

The paper is organized as follows. In section 2 we describe our sensor system. The task of 3-D data acquisition is divided into two subproblems: to see what is illuminated and how to illuminate everywhere. The problem definition of the first subproblem and a proposal to its solution is described in section 3. The second subproblem is represented in section 4. Experimental results are presented in section 5.

2 Sensing

The depth of the scene is measured by the range scanner system [Tsikos, 1989]. It is an active binocular system consisting of a laser, a CCD camera, and a turntable which supports the scanned object. The laser and the camera are coupled. The laser produces a beam which is spread into an illuminating plane. The illuminating plane intersects with the object surface, forming a planar curve (laser-stripe). Each point on the curve is mapped onto a single point in the camera image plane. The distance of the point on the curve to the camera center is determined by the intersection of the illuminating plane and the line which goes through the camera center

*This research was supported in part by AFOSR Grants 88 0244, AFOSR 88-0296; Army/DAAL 03-89-C-0031PRI; NSF Grants CISE/CDA 88-22719, IRI 89-06770, ARPA Grant N0014-88-K-0630 and Dupont Corporation.

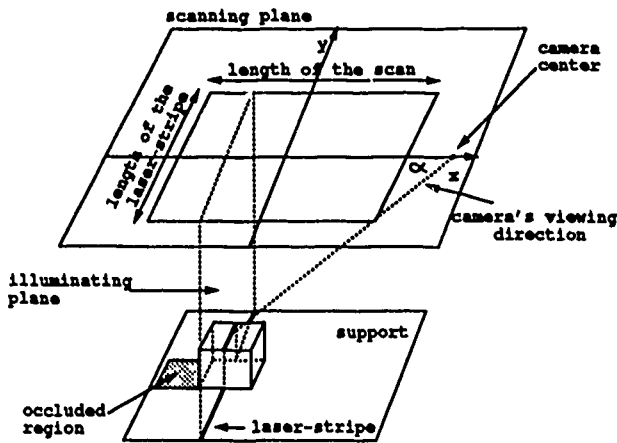


Figure 1: Sensing

and the point on the image plane. The range image is obtained by scanning the object in a series of parallel illuminating planes. This is achieved by moving the scene support with constant velocity perpendicular to the illuminating plane. The y -axis on the image corresponds to the laser-stripe and the x -axis corresponds to the shift of the support. During the scanning, the laser sweeps out a plane, called the *scanning plane* (see fig. 1). The distance measurements are transformed into the perpendicular distance between the scanning plane and the surface of the objects and are stored as intensity values in the final range image. The higher values correspond to smaller distances, the lower to greater distances. The system is calibrated so that the depth of the support plane has the lowest value, which is greater than zero. This enables us to detect the *occluded regions* whose intensity value is zero. Occluded regions arise when the reflected laser light does not reach the camera but is intercepted by some parts of the object. We can construct a complete $2\frac{1}{2}$ -D data of the scene by scanning from different directions (in the same scanning plane). For every pair (x, y) in the scanning plane, we get the distance of the closest point on the surface to the scanning plane (fig. 2),

$$h(x, y) = \max(h_{on\ surface}(x, y)). \quad (1)$$

We generate complete $2\frac{1}{2}$ -D data of the scene from a union of all those views that together eliminate the occluded regions.

3 Getting the complete $2\frac{1}{2}$ -D data of the scene

Definition of the scanning direction. Let the laser stripe and the camera move from left to right where the camera is to the right of the laser-stripe. The *scanning direction* is the angle of counterclockwise rotation of the camera-laser configuration about the origin of the scanning plane. However, in our experiments we rotated the support in clockwise direction instead.

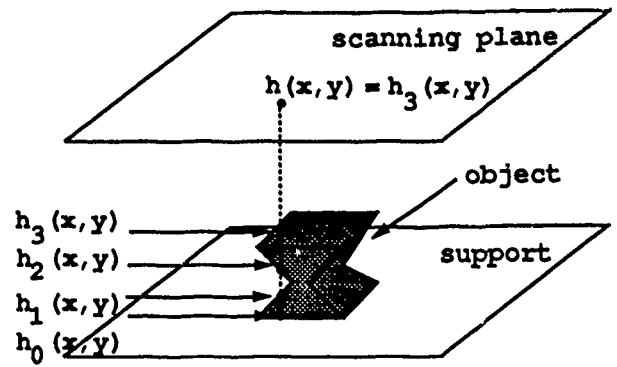


Figure 2: Height at the point (x, y) is defined by the closest point on the surface to the scanning plane

3.1 Finding the next view.

In all occluded regions the information about the height of the surface with respect to the scanning plane is unknown. The question is how to define the next scanning direction to acquire the missing information.

3.1.1 Problem definition

For a given point $p_i(x, y)$ in the scanning plane, we want to find out all the directions from which $p_i(x, y)$ is visible.

In order to determine whether a given point $p_i(x, y)$ is visible from the direction φ_j , we must check if any of the points $p_k(x, y)$ in the scene along this direction (fig. 3) occludes the point $p_i(x, y)$. We consider the relation between the distance l of two points in the scanning plane and the length of occlusion l' produced by $p_k(x, y)$ in direction φ_j and defined as:

$$l' = \frac{h(p_k) - h(p_i)}{\tan(\alpha(h(p_k)))}, \quad (2)$$

where $h(p_i)$ and $h(p_k)$ represent the height of the scene at points $p_i(x, y)$ and $p_k(x, y)$, respectively.

If the distance between $p_i(x, y)$ and each $p_k(x, y)$ is greater than the computed length of occlusion l' , then $p_i(x, y)$ is visible from direction φ_j . In the same scanning plane the point $p_i(x, y)$ can be seen from several different directions $\varphi_j, \varphi_{j+1}, \dots$. To compute the set of all directions from which $p_i(x, y)$ is visible, we have to know the corresponding height for each point in the scanning plane.

As indicated previously, after the first scan, the height of the points that belong to the occluded regions is unknown. To determine the next scanning direction we want to compute the visible directions for all the points in the occluded regions. It follows that we must make some assumptions about the height of these points. The question is, how restrictive assumption about the height of the point in the occluded region can be made. If the assumption is too restrictive then the solution may not be found at all, while if the assumption is too weak then it may give a direction from which the point is not visible.

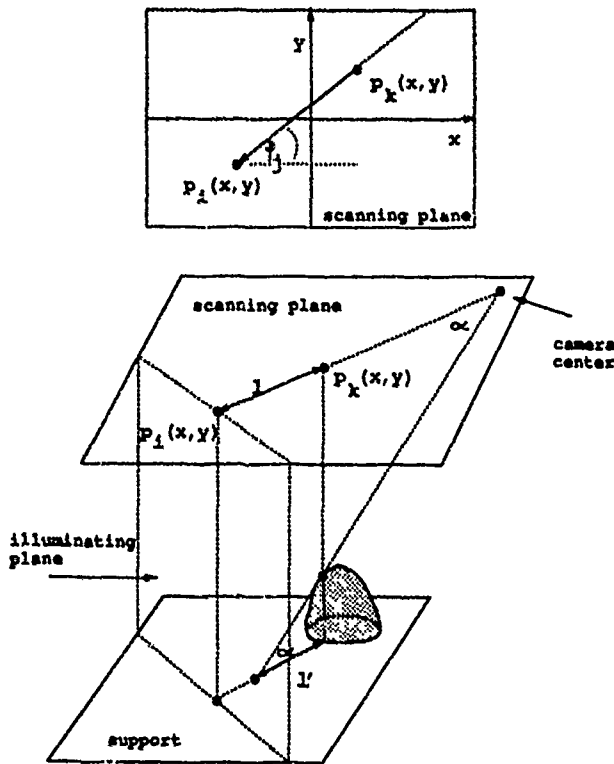


Figure 3: The relation of the distance l between points $p_i(x, y)$ and $p_k(x, y)$ and the length of occlusion l'

3.1.2 Occluded region representation

We approximate the occluded regions by polygons. The next views are computed from the analysis of these polygons and from the height values of their borders.

The contours of the occluded regions in the image are found by Pavlidis' algorithm TRACER [Pavlidis, 1982]. The contours of the occluded areas are represented as contour descriptors $x(s)$ and $y(s)$ for each area separately. Each contour must be segmented into a series of straight lines. The breakpoints on the contour are points of maximum curvature.

Points of maximum curvature

In order to find the points of maximum curvature, the contour descriptors $x(s)$ and $y(s)$ are first smoothed to filter out noise. The derivatives $\frac{dx(s)}{ds}$ and $\frac{dy(s)}{ds}$ are computed to get the tangent angle function as

$$\phi(s) = \tan^{-1}\left(\frac{dy/ds}{dx/ds}\right). \quad (3)$$

Special processing is required to handle phase wrapping. The breakpoints of the contour are obtained from curvature function which is the derivative of the tangent angle function

$$\kappa(s) = \frac{d\phi(s)}{ds}. \quad (4)$$

The positive maxima and negative minima of the curvature function $\kappa(s)$ represent the convex and concave vertices of the polygon.

Border height definition

The height of the contour $h(s)$ is found by searching in a narrow zone around the occluded area. For each pixel on the contour the search is done for n pixels in three directions. The directions of the search are defined by the change in the x and y coordinates of the new pixel on the contour (fig. 4). The highest value found in the search is chosen as the pixel height. The height of the border between two vertices is approximated by a constant. It is defined as the median of all height values on the contour between the two vertices.

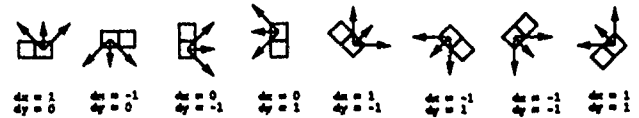


Figure 4: Searching directions for the pixel height.

Polygon representation

The points of maximum curvature $p_i(x, y)$ with $i = 1, \dots, n$ are the vertices of a polygon $P(p_1, p_2, \dots, p_n)$ which approximates the occluded region. The vertices are ordered clockwise so that when one moves in the direction of ordered vertices an occluded region is on the right side of an edge. For each edge $e_i(p_i, p_{i+1})$ in polygon P its angle (fig. 5) with respect to the common coordinate frame is defined by the equation

$$\theta(e_i) = \tan^{-1}\left(\frac{y_i - y_{i+1}}{x_i - x_{i+1}}\right) \quad (5)$$

where (x_i, y_i) and (x_{i+1}, y_{i+1}) are the coordinates of the vertices p_i and p_{i+1} respectively.

3.1.3 Analysis of occluded regions

Two properties are added to each edge according to its angle $\theta(e_i)$ in the common coordinate frame and its height $h(e_i)$ (fig. 7).

1. **Occlusion** If φ is the scanning direction by which the image was obtained then an edge $e_i(p_i, p_{i+1})$ with angle $\theta(e_i) \in (\varphi, \varphi + 180^\circ)_{\text{counterclockwise}}$ is called an *occluding edge* $e_{occ}[i]$ and belongs to the set of occluding edges, O .

If in polygon P we draw lines through the endpoints of the occluding edges in the direction φ , we cut the polygon P into areas A , as shown in figure 6. To each edge $e_{occ}[i]$ in the polygon P belongs such an area A_i . Polygon P can then be defined in the following way

$$P(p_1, p_2, \dots, p_n) = \bigcup_{\{i|e_{occ}[i] \in O\}} A_i, \quad (6)$$

Note that only occluding edges create areas A .

2. **Activity** The second property is derived from the height of the border. Let us say that $e_{occ}[i]$ is an occluding edge to which the area A_i belongs, and e_j is an edge which limits the same area A_i . We compare the length of occlusion l' on the range

COMMON COORDINATE FRAME OF MULTIPLE VIEWS IN ONE SCANNING PLANE

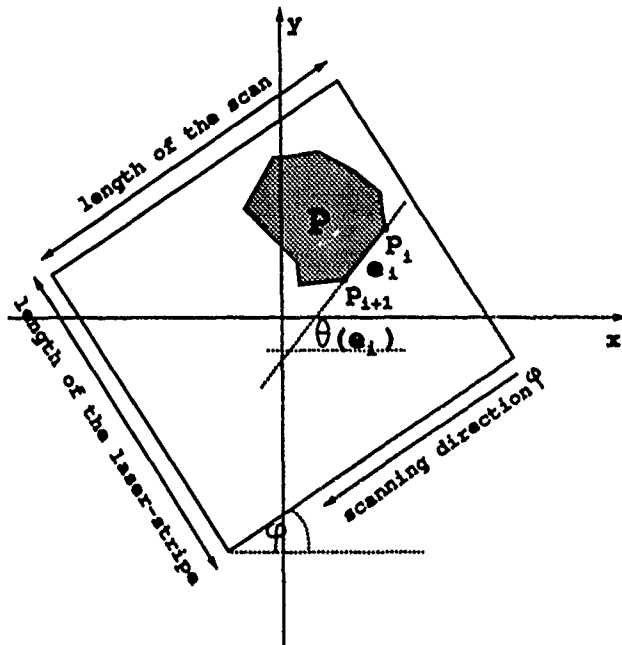


Figure 5: Angle θ of the edge e_i with respect to the common coordinate frame.

image caused by the occluding edge $e_{occ}[i]$ with the occlusion caused by the difference in heights ($h(e_{occ}[i]) - h(e_j)$) for $\alpha(h(e_{occ}[i]))$. For each area A_i , the edges which limit the area are tested as follows:

If e_j limits A_i then

$$e_j = \begin{cases} \text{inactive} & l' \leq \frac{h(e_{occ}[i]) - h(e_j)}{\tan(\alpha(h(e_{occ}[i])))} \\ \text{active} & l' > \frac{h(e_{occ}[i]) - h(e_j)}{\tan(\alpha(h(e_{occ}[i])))} \end{cases} \quad (7)$$

where $j=1, \dots, n$ and l' is measured at the middle point of the edge e_j . An occluding edge is also an active edge. Active edges are these edges of polygon P which are able to occlude the polygon P .

From these properties, we say that the pixels in occluded regions can be visible only from inactive edge. This statement implicitly includes two assumptions:

1. Changes in surface height inside the occluded regions are so small that they cannot cause occlusions.
2. Changes in surface height outside the border of occluded regions are so small that they cannot cause occlusions.

Computing visible directions for each black pixel in the image is computationally expensive. Instead of computing visible directions for each pixel, we compute the visible directions for areas A .

For each area A_i in polygon P , we shall define the viewing angle $\Phi(A_i)$ as the sector containing all directions from which the area A_i can be seen. The viewing angle is computed in two steps:

scanning direction 0°

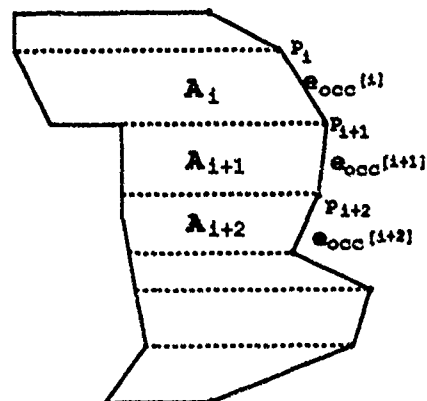
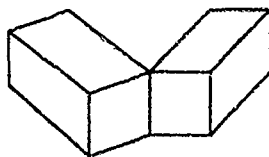
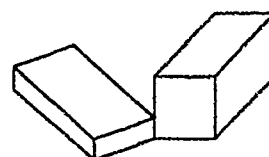


Figure 6: Areas A belonging to occluding edges.

scanning direction 0°



- occluded region
- a occluding edges
- b active edges
- c inactive edges

Figure 7: Definition of edges.

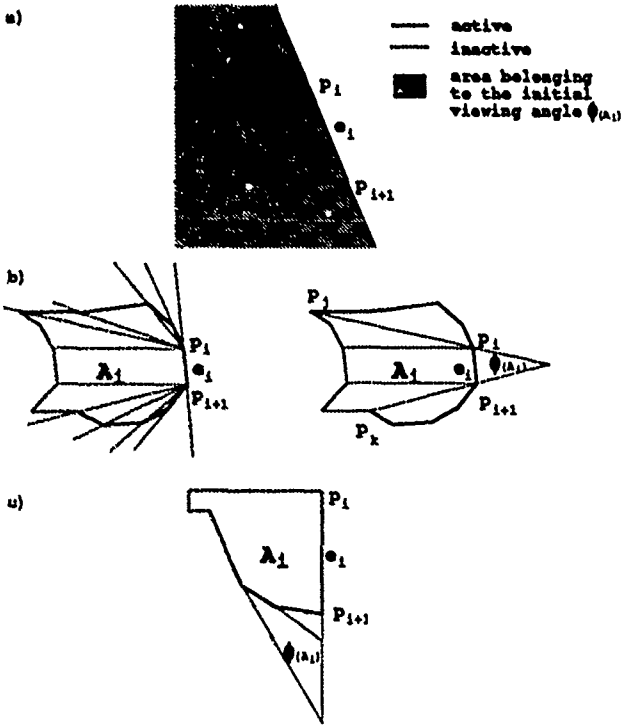


Figure 8: Narrowing the angle $\Phi(A_i)$

1. Initial viewing angles $\Phi(A_i)_{initial}$ are set up.
2. The viewing angles are narrowed by each active edge.

The initial viewing angles definition

The initial viewing angle $\Phi(A_i)_{initial}$ for each area A_i is defined by the angle of the occluding edge $e_{occ}[i]$ the area A_i belongs to:

$$\Phi(A_i)_{initial} = [\varphi(e_{occ}[i]), \varphi(e_{occ}[i]) + 180^\circ]_{counterclockwise} \quad (8)$$

Viewing angles modification.

To each initial viewing angle $\Phi(A_i)_{initial}$ belongs an area which is to the right of the line defined by the occluding edge $e_{occ}[i]$ (fig. 8a). This area can be divided into three parts, an area above A_i , an area in which A_i lies and an area below A_i . Only the edges within these three areas modify the angle:

- Inactive edges do not modify the angle.
- Active edges can modify the initial viewing angle $\Phi(A_i)_{initial}$ in the following way.
 - If the edges e_1, \dots, e_r are active and limit the area A_i , the viewing angle (fig. 8c) is defined as

$$\Phi(A_i) = \Phi(A_i)_{initial} \cap [\theta(e_1), \theta(e_1) + 180^\circ] \cap \dots \cap [\theta(e_r), \theta(e_r) + 180^\circ]. \quad (9)$$
 - If an edge e , is active and is not a part of the border of the area A_i , then the angle $\Phi(A_i)$ is narrowed so that e , is excluded from the area

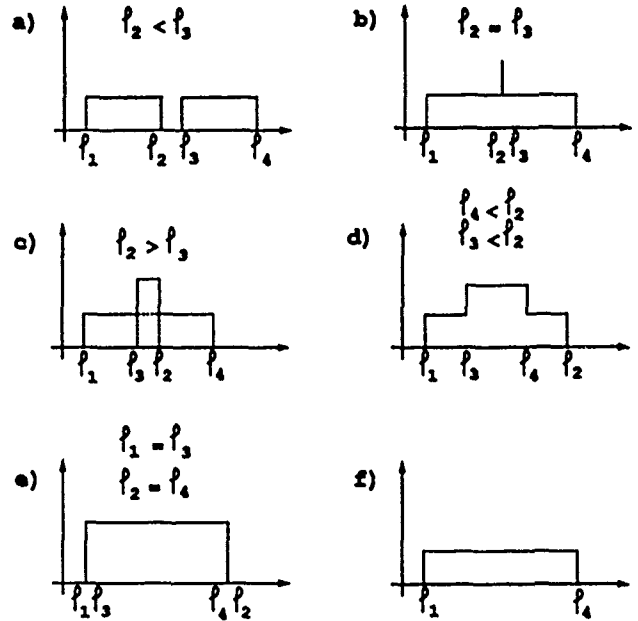


Figure 9: a)-e) possible histograms of two viewing angles, f) impossible histogram

of the viewing angle. The narrowing is done through the endpoints of the active edges on both sides of the area A_i (fig. 8b). The new viewing angle for area A_i is then:

$$\Phi(A_i) = [\varphi_1, \varphi_2]_{counterclockwise}. \quad (10)$$

φ_1 and φ_2 are the lower and upper bound of the viewing angle, defined as

$$\varphi_1 = \tan^{-1}\left(\frac{y_j - y_i}{x_j - x_i}\right), \quad (11)$$

$$\varphi_2 = \tan^{-1}\left(\frac{y_k - y_{i+1}}{x_k - x_{i+1}}\right). \quad (12)$$

(x_j, y_j) , (x_k, y_k) are the coordinates of the vertices $p_j(x, y)$ and $p_k(x, y)$ in the common coordinate frame. Vertex $p_j(x, y)$ is that vertex of the upper area which narrows the viewing angle of the area A_i , the most from above. Similarly, the vertex $p_k(x, y)$ narrows the viewing angle the most from below.

Whenever the viewing angle $\Phi(A_i)$ becomes empty the area A_i can not be seen from one view. In this example the area A_i must be split on smaller parts and new viewing angles must be computed.

3.1.4 Determination of the next scanning direction

After having defined the viewing angles, we must determine the next scanning direction. If there exists a direction from which the whole occluded region can be seen at once then this direction must appear in all viewing angles. We have to intersect the viewing angles

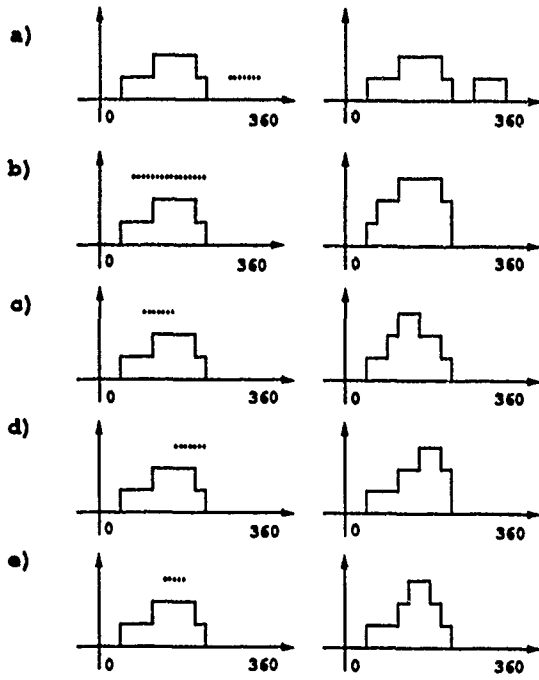


Figure 10: Including a new viewing angle into the histogram

and choose a scanning direction from the global intersection if it is nonempty or from more partial intersections otherwise. To select the scanning directions from the right partial intersections we build a histogram which shows in how many viewing angles a certain direction appears. The histogram can be constructed by successively adding viewing angles and incrementing the count of all directions in added viewing angle. Since the viewing angles are closed sets, two viewing angles, $[\varphi_1, \varphi_2]_{ccw}$ and $[\varphi_3, \varphi_4]_{ccw}$ where $\varphi_1 \leq \varphi_3$, can form the histograms like in (fig. 9 case a,b,c,d,e). The case (fig. 9f) is not possible. Whenever we include a viewing angle into the histogram, there exist the following possibilities:

1. The viewing angle forms a new maximum (Fig. 10a).
2. The viewing angle leaves the width of an old maximum unchanged from the left and from the right side (Fig. 10b).
3. The viewing angle modifies an old maximum from
 - the right side (Fig. 10c),
 - the left side (Fig. 10d),
 - both sides (Fig. 10e).

In the third case above we decrease the maximum's width but increase its height by one which means that maximum remains within all previous viewing angles. In each viewing angle there is at least one maximum. If there is only one maximum in the histogram, it represents the global nonempty intersection. From each direction in the global maximum the whole occluded region can be seen at once. If there is more than one maximum in the histogram and we select one scanning direction from each maximum, then we are able to see the whole occluded region.

Histogram decomposition

Some viewing angles can include more than one maximum which means that some areas A will be seen more than once if we take a scanning direction from every maximum. We must test if views from *all* maxima are really necessary. For each viewing angle we must find the number of maxima it includes. If in viewing angle $\Phi(A_i)$ there exists only one maximum then the area A_i can be seen only from this maximum, so the scanning direction from this maximum is necessary. We label all such maxima and remove from the histogram all viewing angles which include at least one labeled maximum. From the remaining viewing angles we construct a new histogram and repeat the procedure until all viewing angles are removed. Then the new scanning directions must be selected from all labeled maxima.

3.2 Combining images from different viewing positions in one scanning plane into one coordinate frame

The images obtained from the previously selected scanning directions are combined into one coordinate frame where the largest pixel value is kept as the pixel height.

The coordinates (x_i, y_i) of the pixels in the range image obtained by the scanning direction φ are transformed into the coordinates (x_c, y_c) of the common coordinate frame by the following transformation:

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad (13)$$

Because of imprecise sensing and inaccuracy of the transformation parameters the final merging is achieved by correlation;

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} + \begin{bmatrix} \cos(\varphi + \Delta\varphi) & -\sin(\varphi + \Delta\varphi) \\ \sin(\varphi + \Delta\varphi) & \cos(\varphi + \Delta\varphi) \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}, \quad (14)$$

where Δx , Δy , and $\Delta\varphi$ are discrete values from the intervals $[-X, X]$, $[-Y, Y]$, and $[-\phi, \phi]$, respectively, which give the maximum correlation between the image of the common coordinate frame and the current transformed image.

The height at pixel (x_c, y_c) is the maximal height of all pixels which are transformed into it.

3.3 Completion of the analysis of one plane

To compute the new scanning directions from the first view we assume that the changes in surface height in the occluded regions are too small to cause occlusions. The same is assumed for the regions outside the border of the occluded regions. If this is not the case, and after combining images from different viewing positions into one coordinate frame, there still exist occluded regions, they should be explored from other directions. Since

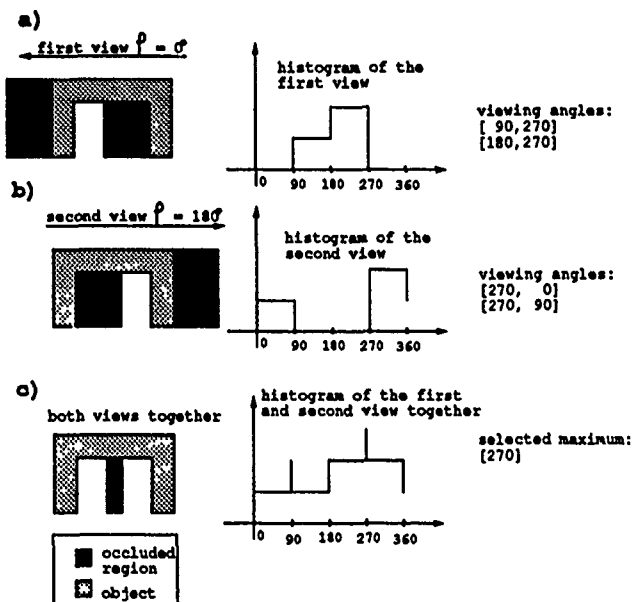


Figure 11: Example 1

we use only the height information of the border of occluded regions to compute the viewing angles, we may end up with too wide viewing angles. When we compute the next scanning direction it is important to use the information about the active borders obtained by the previous scan. In our algorithm, we incorporate this information by building a new histogram from the previous one. Let us illustrate this by two examples.

Example 1

In the image of the first scan we find two occluded regions (fig. 11). From their borders the viewing angles are computed. The viewing angles are $[90^\circ, 270^\circ]$ for the larger region and $[180^\circ, 270^\circ]$ for the smaller region. The viewing angle of the smaller region is too wide, because we do not check the visible region outside the border of the occluded regions. In the histogram we get one maximum at $[180^\circ, 270^\circ]$. If we select 180° as the second scanning direction we still have an invisible part after merging both images. The maximum of the histogram of the second view is again not restrictive enough. By combining both histograms, we get two maxima. Since all viewing angles include the maximum 270° that one is selected as the next scanning direction.

Example 2

In the image of the first view (fig. 12) we find one occluded region. The computed viewing angle is $[90^\circ, 270^\circ]$. After the second view we detect a low box inside the occlusion of the first scan. If in the second view the occluded region of that box does not reach the larger box, the histogram of the second view does not give the correct solution. By incorporating the first histogram into the second histogram, we get three maxima. We do not select a scanning direction from the middle maximum $[-45^\circ, 45^\circ]$ because it includes the scanning

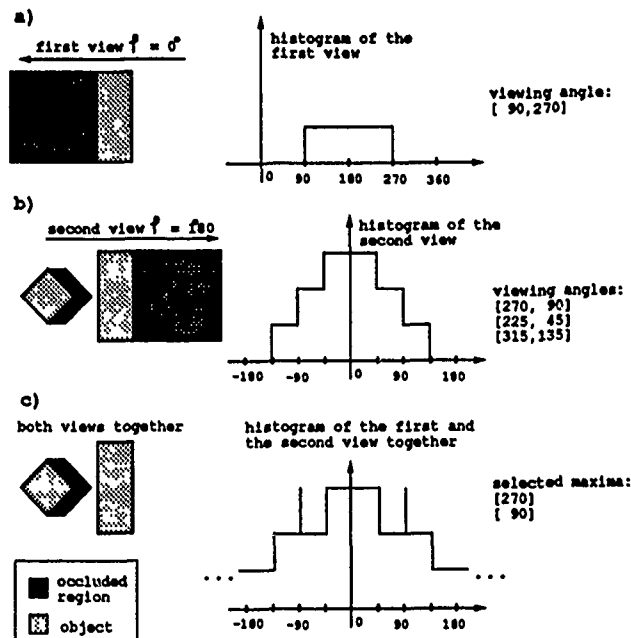


Figure 12: Example 2

direction $\varphi = 0^\circ$ of the first scan.

In general the computation of the next view is completed in the following way. For each scanning direction that was taken we compute corresponding viewing angles. A single histogram is then constructed from both the old one and new viewing angles. New scanning directions are computed in a similar way as in subsection 3.1.4. During the histogram decomposition we do not remove the viewing angles of the old histogram. We do not select the scanning direction from those maxima which include the scanning directions already taken. A possibility exists where there is no viewing angle which includes only one maximum. In this case, a histogram decomposition must be done for each local maximum. The solution with the minimum number of necessary scanning directions is selected. The procedure stops either when all occluded regions are removed or when from all maxima scanning direction have been taken.

4 Searching for the next scanning plane.

In the previous section we described how to rotate the sensor system in one plane to get the complete $2\frac{1}{2}$ -D data of the scene. In each scan we illuminate the same part of the scene surface but we see its different parts. The next problem is how to orient the illuminating plane to illuminate the whole surface. We would like to solve this problem on the base of the $2\frac{1}{2}$ -D information we already have.

The points on the surface to which the illumination is tangent (fig. 13) form borders which separate illuminated from non-illuminated surfaces. These borders are *occluding borders*. An occluding border makes a projection on the other parts of the surface in direction of illumina-

tion. The surfaces, which are spread between occluding borders and their projections, must be illuminated.

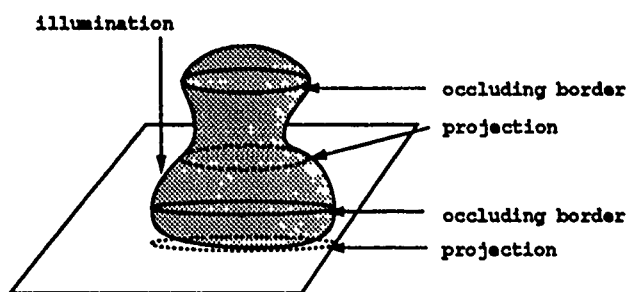


Figure 13: Occluding borders, and their projections

Assume that the scanning plane in which we constructed the $2\frac{1}{2}$ -D data is the upper basic face of a cylinder (fig. 14). If the new scanning plane can only be perpendicular to the first one, it can be defined by an angle δ , which is the angle of rotation of the illuminating plane about the origin of the upper basic face of the cylinder. From the occluding borders we would like to compute the rotation angles δ of the next scanning planes.

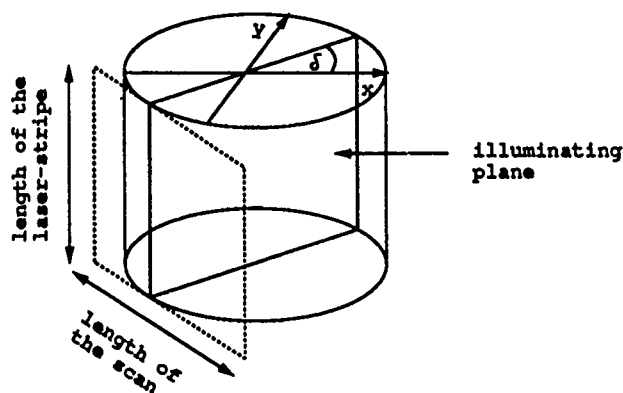


Figure 14: Orientation of the illuminating plane defined by angle δ

4.1 Occluding borders

Occluding borders are defined by the jumps in height in the $2\frac{1}{2}$ -D image of the first scanning plane. With an edge operator which is sensitive to C0 discontinuities, we can locate occluding borders. Occluding borders can be approximated by piecewise linear segments l_i with $i = 1, \dots, m$. In our experiments we use the algorithm proposed by R. Nevatia and K. Ramesh Babu [1980]. In their algorithm, line finding consists of the following steps: determining edge magnitude and direction by convolution of an image with a number of edge masks, thinning and thresholding these edge magnitudes, linking the edge elements based on proximity and orientation, and approximating the linked elements by piecewise linear segments.

Each line segment l_i represents, at the same time, the part of occluding border and its projection. To compute

the next scanning plane we will assume that the surface which is spread between the occluding line segment l_i and its projection, is a flat face F_i . In a similar way, as we did in the previous section, we would like to define for each line segment l_i the illuminating angle $\Psi(F_i)$ as a set of all direction from which the face F_i can be illuminated.

4.1.1 The height of occluding border and the height of its projection

The height of occluding border $h_b(l_i)$ and the height of its projection $h_p(l_i)$ are found by searching in a narrow zone left and right of the linked edge elements. The direction of the search is defined by the change in x and y coordinates of the new edge element in the link. The directions of the search on the left side are the same as those in figure 4 where the searching directions on the right side are just opposite. For each edge element, we calculate a left height taking the median of the height values found in the search on the left. Similarly, we do the same on the right for each edge element. The left and right height of the line segment l_i are approximated by constants which are defined as medians of all height values, between break points p_i and p_{i+1} , which are found on the left and on the right side from the edge elements in the link, respectively. The higher value corresponds to the height of occluding border $h_b(l_i)$ and the lower to the height of its projection $h_p(l_i)$.

4.1.2 An angle of a line segment

The endpoints p_i, p_{i+1} of the line segment l_i are ordered such that the higher surface is always on the right side going in direction from p_i to p_{i+1} . The angle of the line segment l_i in the common coordinate frame of the first scanning plane is

$$\psi(l_i) = \tan^{-1} \left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i} \right) \quad (15)$$

Since the higher surface is on the right side of the line segment l_i the face F_i can be illuminated only from directions in the range $[\psi(l_i), \psi(l_i) + 180^\circ]_{ccw}$.

4.2 Illuminating angles computation

To compute the illuminating angle $\Psi(F_i)$ for each face F_i we must locate the areas (islands) I in the $2\frac{1}{2}$ -D image of the first scanning plane which are higher than the height of projection $h_p(l_i)$ of the occluding line segment l_i . We find the contours for each l_i of such islands I with the same TRACER algorithm as we found the contours of occluded regions in the previous section. These islands represent the obstacles for the light. We must find all these directions from which the light reaches the face F_i without collisions with the islands. To find a solution to this problem we use the work of G.T.Toussaint and J.R.Sack [1983] which has solved the problem of moving polygons in the plane.¹ The line, on which lies the line segment l_i , separates the first scanning plane into two parts. If on the left side of the line there is no such island I which is higher than $h_p(l_i)$, then the face F_i

¹The authors answered the following question: Given a direction d can the polygon P be translated in an arbitrary distance in direction d without colliding with polygon Q .

can be illuminated from any direction in $[\psi(l_i), \psi(l_i) + 180^\circ]_{ccw}$. If there is an obstacle island, then each line lc_j on face F_i connecting the point p_j on the occluding line segment l_i and its projection p_{p_j} (fig. 15a) can be seen from a different subset of directions from the range $[\psi(l_i), \psi(l_i) + 180^\circ]_{ccw}$. The illuminating angle of line lc_j for K obstacle islands is:

$$\Psi(lc_j) = [\Psi(l_i), \Psi(l_i) + 180^\circ]_{ccw} - (\cup_{k=1}^K \alpha_j(I_k)). \quad (16)$$

The angle $\alpha_j(I_k)$ includes all directions from which the island I_k occludes the line lc_j . If we insist upon illuminating the whole face F_i at the same time, then the face can be illuminated, if the intersection between the illuminating angle $\Psi(lc_i)$ (of the line connecting the first endpoint on the line segment and its projection) and the illuminating angle $\Psi(lc_{i+1})$ (of the line connecting the second endpoint and its projection) is not empty. If the intersection is empty, then the face F_i cannot be illuminated at once. The solution can be found by splitting the line segment l_i and respectively the face F_i into two parts and computing the illuminating angle for each part separately (fig. 15b). In the drawing on figure 15a, each line lc_j on the face F_i is occluded from all directions inside the angle α_j . If the $\psi(l_i)$ is the angle of the line segment l_i , then the whole face F_i can be illuminate from any direction in the range $[\psi(l_i), \psi(l_i) + 180^\circ]_{ccw} - (\alpha_i \cup \alpha_{i+1})$. In the illustration on figure 15b the line defined by the first endpoint p_i and its projection p_{p_i} can be illuminated from any direction in the angle β_i . Also, the line defined by the second endpoint p_{i+1} and its projection $p_{p_{i+1}}$ can be illuminated from any direction in the angle β_{i+1} . Since the intersection $\beta_i \cap \beta_{i+1}$ is empty, the face F_i belonging to the line segment l_i cannot be illuminated from only one scanning plane. By splitting the line segment l_i into two parts, the solution is found. The face F_i must be illuminated from the two planes. The orientation δ of the first scanning plane must be selected from the angle γ_i , and the orientation δ of the second scanning plane from the angle γ_{i+1} .

4.3 Computation of the next scanning plane

After computing the illuminating angles $\Psi(F_i)$ for all faces F_i , the angles δ of the next scanning planes are determined from the histogram in the same way as we compute the new scanning directions in the previous section. The directions in the necessary histogram maxima are the initial candidates for rotation angles δ for the next scanning planes.

5 Experimental Results

We tested our algorithm on a number of different scenes. Results are shown for two different scenes. The range images were scanned using a structured lighting laser-scanner with approximately 1mm/pixel spatial resolution and 1.2mm depth resolution.

Scene 1: The scene consists of four boxes of different heights and different sizes. The first view is done in direction 0° . In the image of the first view (fig. 16) the highest box partially occludes the lowest box. The other three boxes are arranged so that they occlude only the

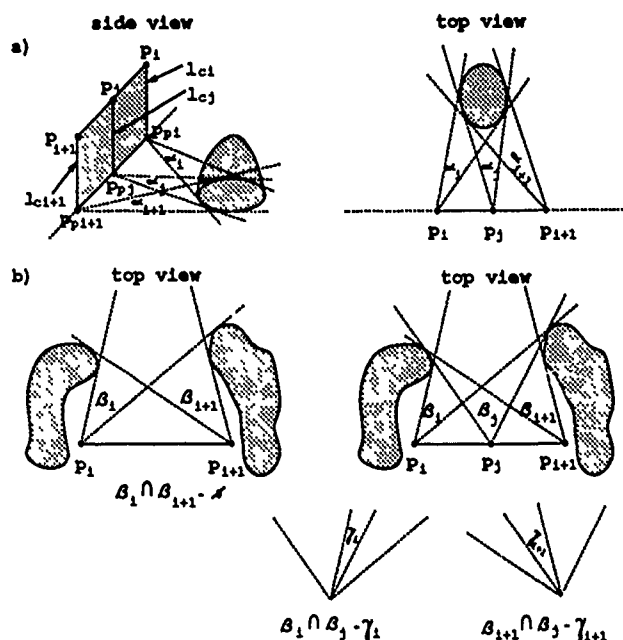


Figure 15: Illuminating angle computation

support. The contour of the occluded region whose intensity value is 0 (black area) is represented in the image of figure 17. From the curvature function of the contour of the occluded region (fig. 18), the polygonal approximation is computed (fig. 19). By searching around the contour in the belt of 7 pixels wide, the border height is obtained (fig. 20). The height of the border between two vertices is approximated by a constant which is the median of all height values between the vertices (fig. 21). From the angle of the edge and from its height, the type of the edge is defined (fig. 22, 23, 24). After computing the viewing angles, the histogram is built (fig. 25). It has two maxima $[129^\circ, 144^\circ]$ and $[157^\circ, 188^\circ]$. We need two additional views to get the complete $2\frac{1}{2}$ -D representation of the scene. We can select as a next scanning direction any direction from the range of both maxima. Since the maximum $[157^\circ, 188^\circ]$ is wider and so more stable, the next view is selected from it. The selected direction is 180° . The third direction can be selected from the lower maximum. If we remove all viewing angles which include the direction 180° from the histogram, we can build a new histogram (fig. 26) where we get wider range $[90^\circ, 144^\circ]$ for the third view. The third selected scanning direction is 120° . In figures 27 and 28, there are the images of both selected directions. By merging the images of the first and the second view we get the image in figure 29. The image of all three views in the common coordinate plane is in figure 30. To compute the rotation angles δ for the next scanning planes from the side, we first locate edges in the image of all three views (fig. 31). The edges represent the occluding borders which are approximated by piecewise linear segments (fig. 32). For each line, the illuminating angle is computed and the histogram of illuminating angles is constructed (fig 33). In the histogram there are seven maxima.

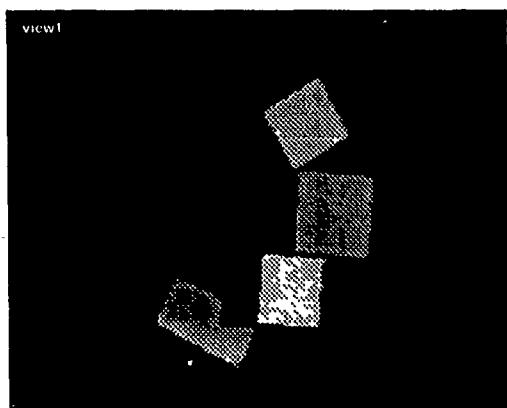


Figure 16: The first view

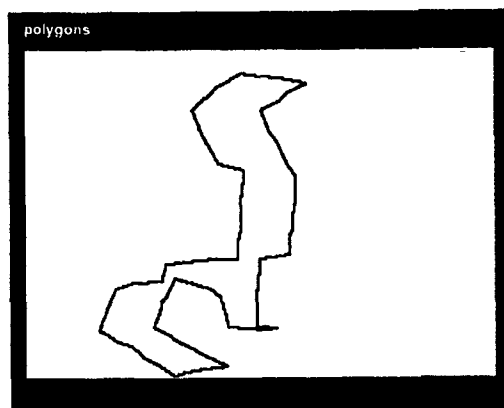


Figure 19: The polygonal approximation of the occluded region

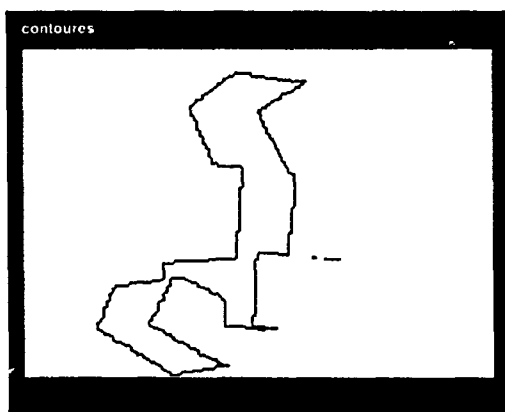


Figure 17: The contour of the occluded region

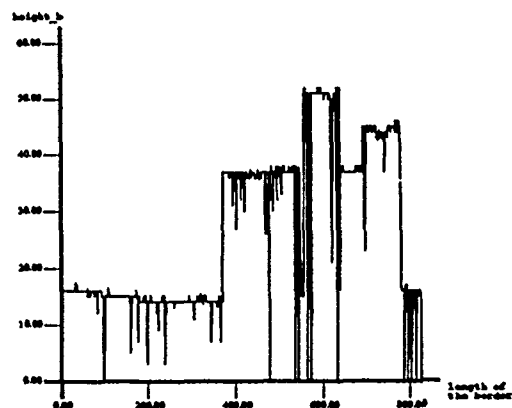


Figure 20: The height of the border of the occluded region

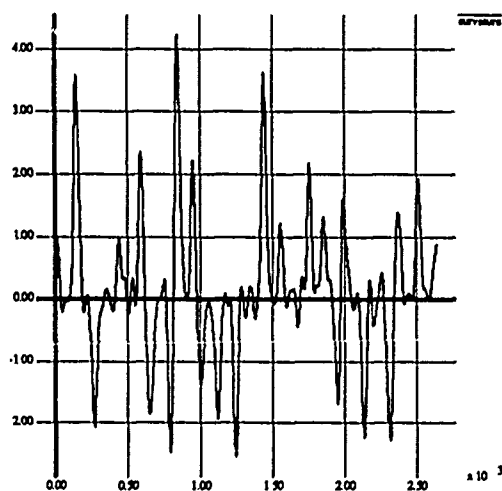


Figure 18: The curvature of the contour of the occluded region



Figure 21: The height approximation of the border of the occluded region

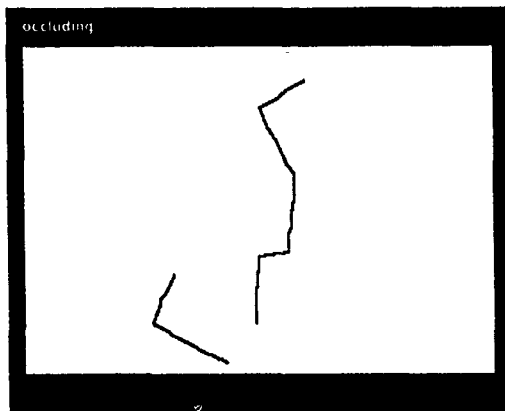


Figure 22: Occluding edges

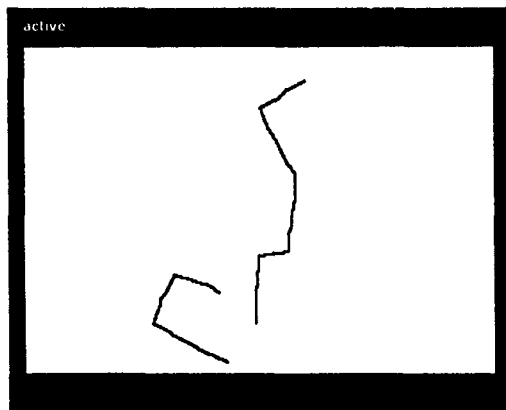


Figure 23: Active edges

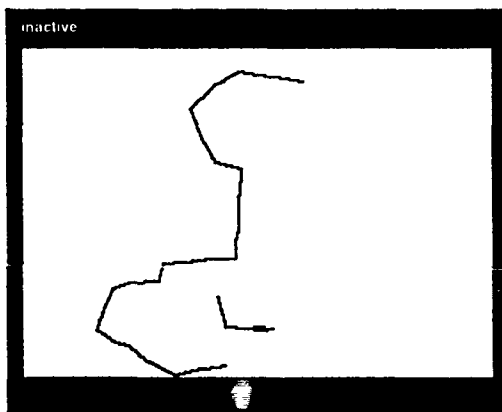


Figure 24: Inactive edges

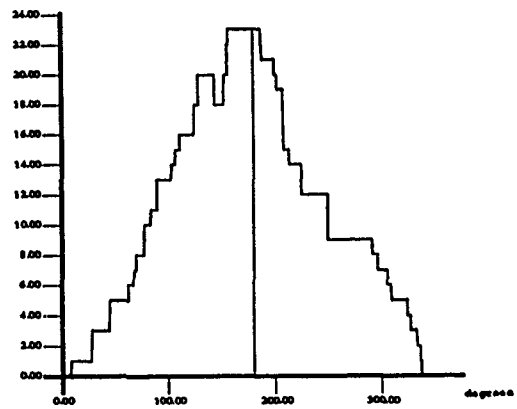


Figure 25: Histogram of viewing angles

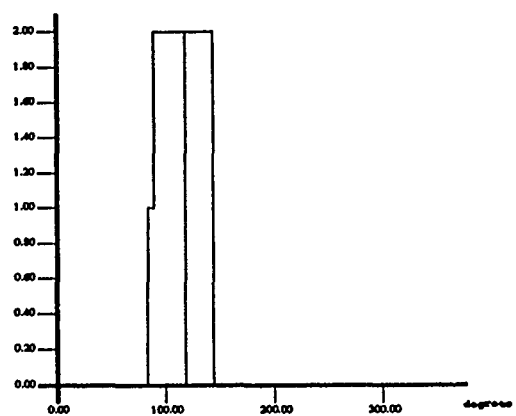


Figure 26: Histogram

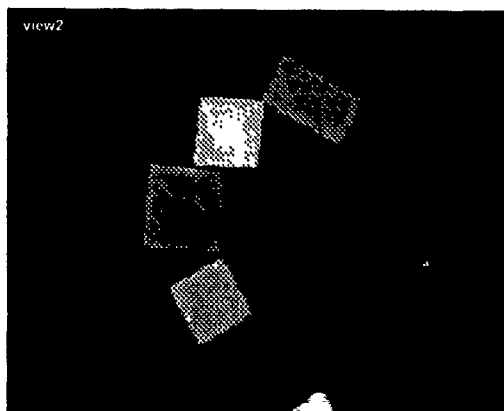


Figure 27: The second view

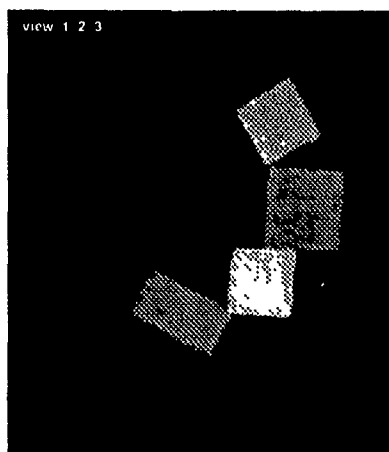


Figure 30: The first, second, and third views in the common coordinate frame

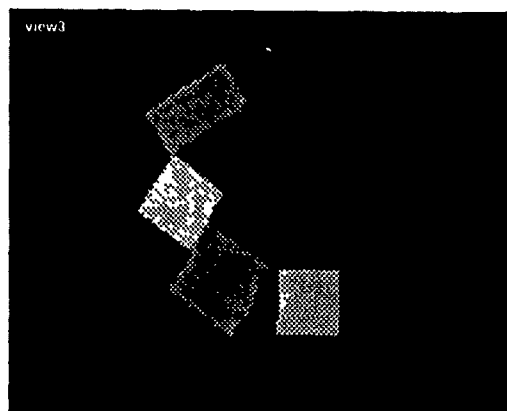


Figure 28: The third view

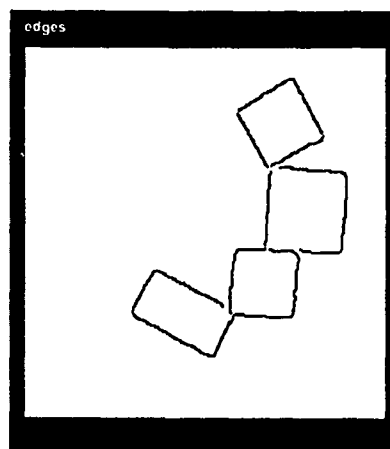


Figure 31: Edges

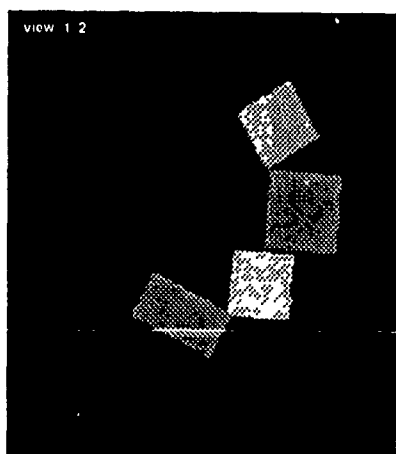


Figure 29: The first and second views in the common coordinate frame

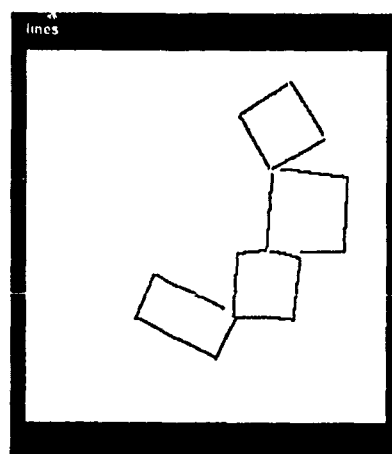


Figure 32: The linear approximation of the edges

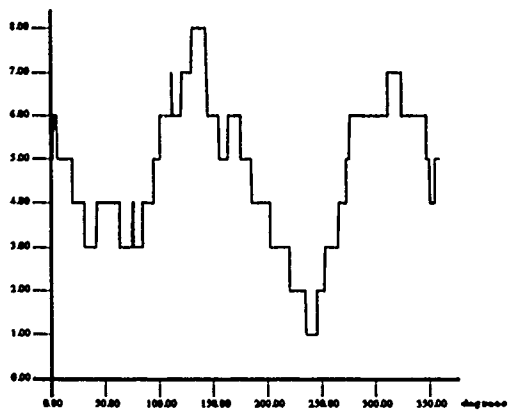


Figure 33: Histogram of illuminating angles

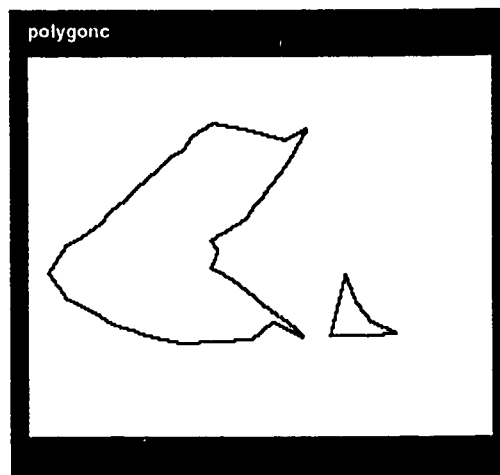


Figure 36: The polygonal approximation of the occluded regions

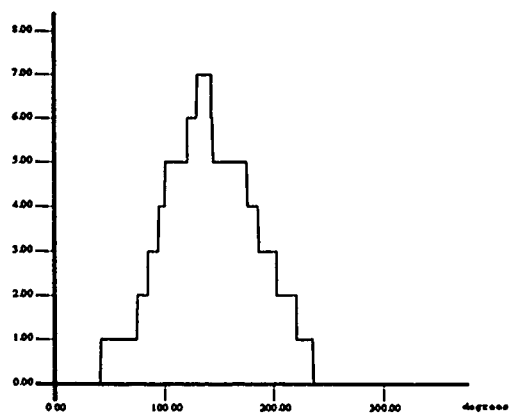


Figure 34: Histogram of illuminating angles after the decomposition

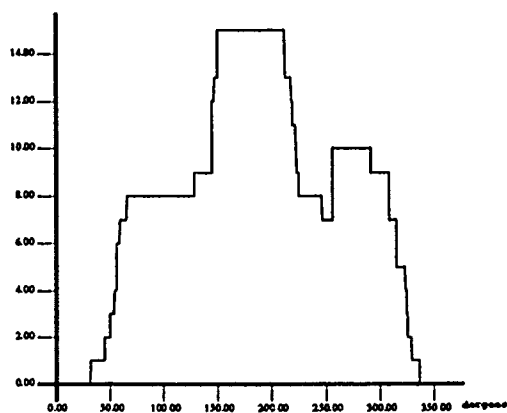


Figure 37: Histogram of viewing angles

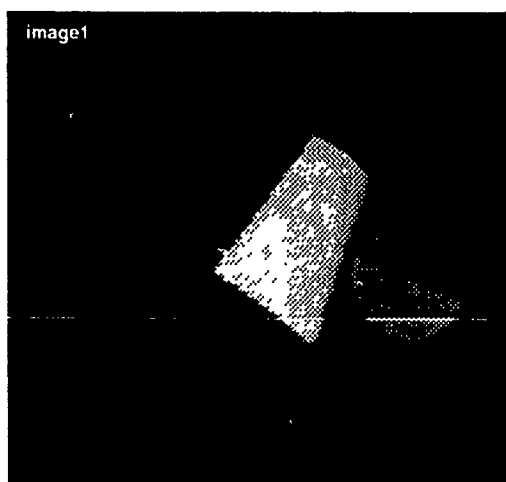


Figure 35: The first view - scene 2

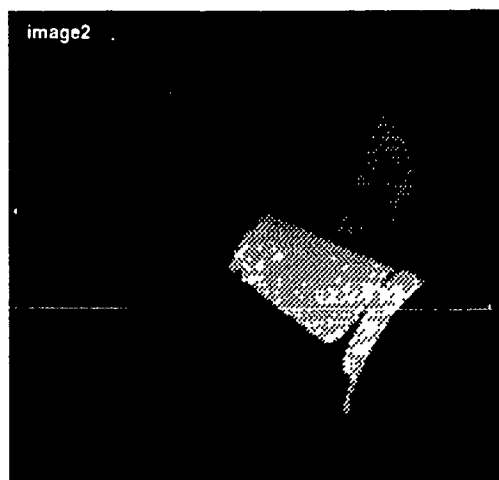


Figure 38: The second view



Figure 39: The third view

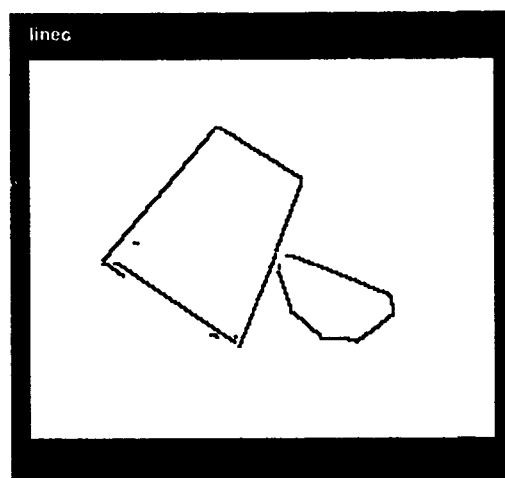


Figure 42: The linear approximation of edges

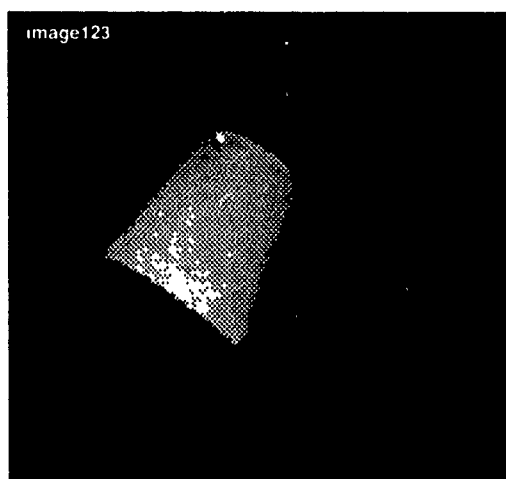


Figure 40: The first, second, and third views in the common coordinate frame

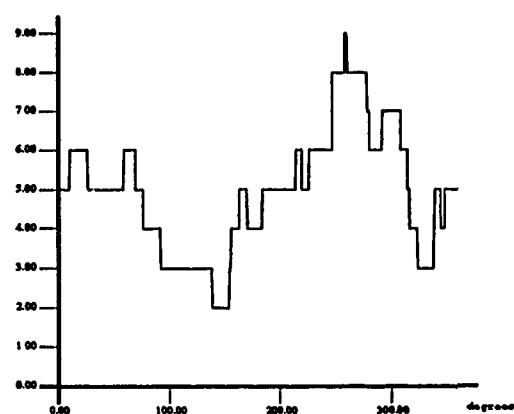


Figure 43: Histogram of illuminating angles

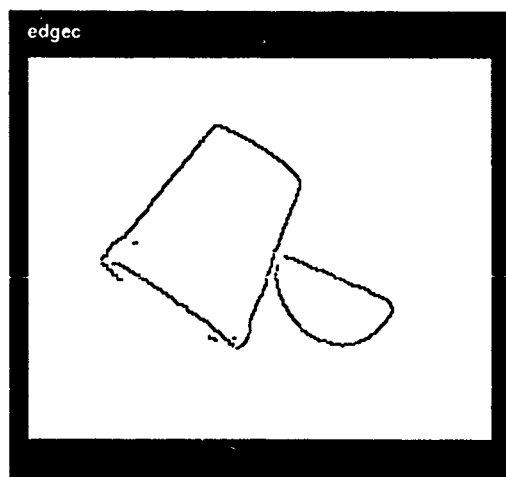


Figure 41: Edges

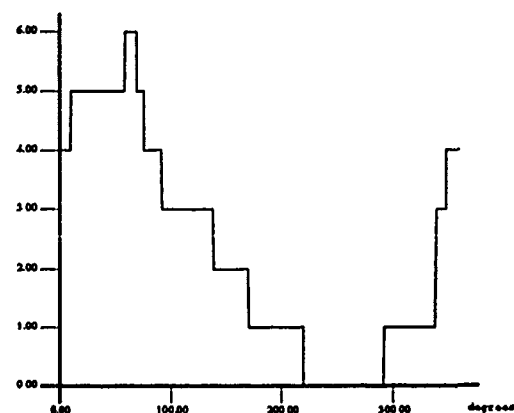


Figure 44: Histogram after the decomposition

During the histogram decomposition, two necessary maxima are found $[2^\circ, 5^\circ]$ and $[312^\circ, 324^\circ]$. From the illuminating angles which do not include the necessary maxima, the new histogram is built. In the second histogram (fig. 34) we have only one maximum $[130^\circ, 144^\circ]$. From all three maxima the rotation angle δ for the next scanning planes must be selected.

Scene 2: The scene consists of a cup and a low geometrical object which represents a half of a cylinder (fig. 35). The first view is done in direction 0° . From the polygonal approximation of occluded regions (fig. 36) the histogram of viewing angles is constructed (fig. 37). The histogram has two maxima: $[150^\circ, 212^\circ]$ and $[256^\circ, 291^\circ]$. Two additional views are done from directions 270° (fig. 38) and 180° (fig. 39). All three views are then merged in the common coordinate frame (fig. 40). The complete $2\frac{1}{2}$ -D image is obtained. From the edges (fig. 41) their linear approximation are made (fig. 42). For each linear segment the areas which are higher than the height of its projection are located and the illuminating angle is computed. From the illuminating angles, a histogram is constructed (fig. 43). During the histogram decomposition one necessary maximum is found $[258^\circ, 260^\circ]$. From the illuminating angles which do not include the maximum $[258^\circ, 260^\circ]$ the second histogram is built (fig. 44) which has only one maximum $[59^\circ, 69^\circ]$. The scene must be illuminated from the side from two scanning planes. The angles δ must be selected from the intervals $[258^\circ, 260^\circ]$ and $[59^\circ, 69^\circ]$.

6 Conclusions

We have developed an algorithm for selecting the proper views to produce the complete $2\frac{1}{2}$ -D data of the scene then from this complete $2\frac{1}{2}$ -D data of the first scanning plane we compute the next scanning planes for 3-D data acquisition. Both planar and curved surfaces are treated in a uniform manner. For the construction of $2\frac{1}{2}$ -D data, only the information in a narrow zone around the occluded regions are used. The holes in the surface which are occluded to the camera from all directions in the first scanning plane cannot be seen because of geometrical properties of the sensor system. To compute the next scanning planes we exploit the occluding borders which are located by an edge operator in the image of first scanning plane. In planning the next scanning planes, we have limited ourselves to the planes which are perpendicular to the first one.

Research is continuing on solving the problem of planning the next scanning planes without limitation.

Acknowledgements

The authors wish to thank Ulf Cahn von Seelen and Howie Choset for comments on an earlier draft of the paper.

References

[Ahuja and Veenstra, 1989] N.Ahuja and J.Veenstra, "Generating Octrees from Object Silhouettes in

Orthographic Views," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, pp. 137-149, Feb. 1989.

[Bajcsy, 1988] R.Bajcsy, "Active Perception," *Proceedings of the IEEE*, August 1988

[Krotkov and Kories, 1988] E.Krotkov, R.Kories, "Integrating Multiple Uncertain Views of a Static Scene Acquired by an Agile Camera System," University of Pennsylvania, Tech. report MS-CIS-88-11, GRASP LAB 135, 1988.

[Nevatia and Babu, 1980] R.Nevatia and K.R.Babu, "Linear Feature Extraction and Description," *Computer Graphics and Image Processing*, Vol.13, 1980, pp. 257-269.

[Pavlidis, 1982] T.Pavlidis *Algorithms for Graphics and image processing*, Computer science press, 1982

[Sakane et al., 1987a] S.Sakane, M.Ishii, and M.Kakikura, "Occlusion avoidance of visual sensors based on a hand-eye action simulator system:HAVEN," *Advanced Robotics*, Vol.2, No.2, pp. 149-165, 1987.

[Sakane et al., 1987b] S.Sakane, T.Sato, and M.Kakikura, "Model-based planning of visual sensors using a hand-eye action simulator system:HAVEN," *Proceedings of the 3rd International Conference on Advanced Robotics ICAR'87*, pp. 163-174, 1987.

[Solina and Bajcsy, 1990] F.Solina and R.Bajcsy, "Recovery of Parametric Models from Range Images: The Case for Superquadrics with Global Deformations," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, pp. 131-147, Feb. 1990.

[Toussaint and Sack, 1983] G.T.Toussaint and J.R.Sack, "Some New Results on Moving Polygons in the Plane," in *Proc. Robotics Intelligence and Productivity Conference*, Detroit, Michigan, November 18-19, 1983.

[Tsikos, 1989] C.J.Tsikos *Laser Range Imaging System User's Guide*, 1989.

Task Oriented Vision¹

Katsushi Ikeuchi and Martial Hebert

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

ABSTRACT

This paper proposes a task-oriented approach to the design of vision systems. The approach changes the architecture of a vision system depending on each task. Toward establishing a systematic methodology for task-oriented vision, this paper overviews two recently completed vision systems, analyzes their design philosophy, and derives a general framework for the task-oriented vision.

1 Introduction

A critical issue in designing a model-based vision system is to organize the relevant components into a vision program. Such components include object representations, features, segmentation methods, and image acquisition sensors. The choice of such vision components governs the quality of the resulting vision programs.

There are two approaches to the problem of organizing the components: general purpose oriented and task-oriented. Researchers in the general purpose oriented school claim that we should build the vision system to solve all the vision tasks using a single architecture, that is, an architecture in which a fixed selection of components is executed in a fixed order. On the other hand, researchers in the task oriented school claim that we should prepare different architectures of vision systems and that, depending on the task, we should change an architecture by using an analysis of the task specifications to systematically select the vision components.

We believe that the task oriented approach is more reasonable in the current stage of computer vision. In the task-oriented approach, however, few attempts have been made to clarify and establish a methodology to choose appropriate components among available ones beyond several ad hoc selections.

This paper proposes a task-oriented vision approach and investigates the design of vision systems in a systematic way. We will focus on the design of robotics systems that involve the

localization and grasping of objects because it is a good illustration of the task oriented approach. In order to illustrate this approach, this paper will overview two vision systems recently completed: Rock Sampling Vision for planetary rover robots and Bin Picking Vision for industrial robots. We will illustrate the task oriented approach in designing these two vision systems. Then, we will derive a general framework for designing vision systems under the task-oriented approach.

2 Rock Sampling System

One of the most important goals of a planetary exploration mission is to collect and analyze terrain samples. As part of the CMU Ambler project [2], we are investigating techniques to collect small rocks in sand. This section overviews the architecture of the rock sampling system.

Image Acquisition Sensor Figure 1(a) shows a typical scene to the system. From this scene, the range image, as shown in Figure 1(b), is acquired using a range finder [19].

Segmentation From a range image, three types of features are extracted:

- shadows
- orientation discontinuities
- range discontinuities

Figure 1(c) shows the features extracted from the scene.

These features give an indication of where the boundaries of the rocks may be located in the scene. These features are, unfortunately, not sufficient for reliably extracting rocks from the scene, because the rocks may be small or partially buried in the sand. Therefore, we cannot use a simple region extraction technique that would assume that the features are grouped into closed boundaries. Instead, we implemented an iterative rock finding algorithm, similar to the "snake" algorithm [16], that grows a closed contour until it fits to the boundary of each rock.

The boundary of each shadow region is taken as a rock hypothesis. Since we know the configuration of the sensor, we can derive the approximate location of the center of a rock from the distribution of the corresponding shadow region. The snake is initialized as a one-pixel contour at the hypothesized center. It is then iteratively deformed until it closely approximates the shape of the rock. Figure 2(a) shows the outline of our segmentation algorithm.

¹This research was conducted in the Intelligent Modeling Laboratory, the Robotics Institute, Carnegie Mellon University. Image Understanding Research in the Intelligent Modeling Laboratory is supported in part by NASA under Grant NAGW 1175, and in part by the Defense Advanced Research Project Agency, DOD, through ARPA order No. 4976, and monitored by the Air Force Avionics Laboratory under contract F33615-87-C-1499. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, NASA, the Defense Advanced Research Project Agency or the U.S. Government.

We implemented the snake using two kinds of energy field: external and internal. The external energy field is given by the interaction between the snake shape and external features such as shadow. The internal energy field is given by the snake shape itself. More precisely, the external energy is given by the following three attractive forces:

- shadow attracter
- orientation discontinuity attracter
- range discontinuity attracter

Following the attractive forces, the boundary moves towards the surrounding features.

The internal energy field tries to maintain the reasonable shape of the snake. We implemented the following internal energy field:

- center attracter
- region attracter

A snake is attracted by its center position. It is also attracted by itself; a snake tries to form a compact shape.

A snake terminates its expansion when all the forces are in equilibrium. Figure 2(b) shows the rock region, shown in blue, which has been extracted by this segmentation algorithm. This approach allows us to locate rocks in the scene even when only a very small number of visual features are extracted from the image. This departs from other vision systems which implicitly assume that strong and reliable features can always be extracted, and therefore would not perform well in the type of unstructured environment that we are considering.

Representation Once a region corresponding to a rock has been extracted by the snake segmentation method, the corresponding set of 3-D points is approximated by a superquadric surface. A superquadric is a generalization of an ellipsoid that can represent a wide variety of shapes with a small number of parameters [18, 1].

We chose superquadrics as our representation for two reasons:

- Superquadrics are appropriate for blob-like shapes.
- Fitting superquadrics to a set of points from a partially visible object gives an estimate of the whole shape of the object, whereas more local surface representations would provide a representation of only the visible part of the object.

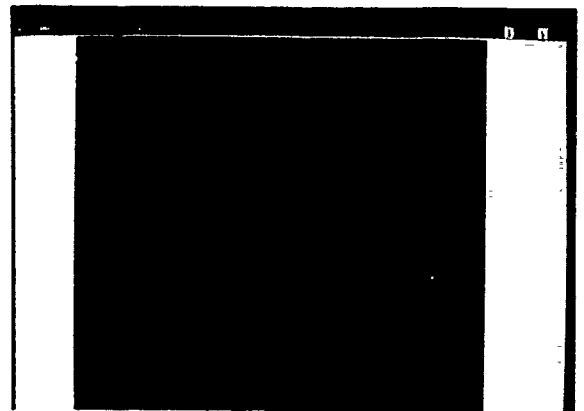
The superquadric fitting algorithm is a standard gradient descent on the parameters of the surface as described in [20]. Figure 3 shows a superquadric representation of the rock.

Grasping Strategy The superquadric fitting module provides the following parameters of a rock:

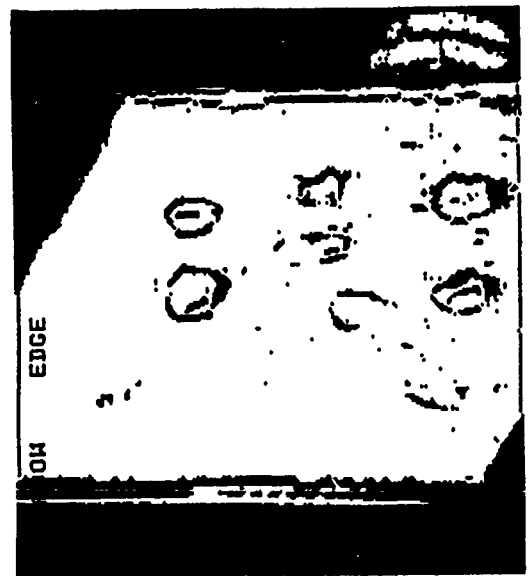
- mass center position
- size
- axis direction



(a)

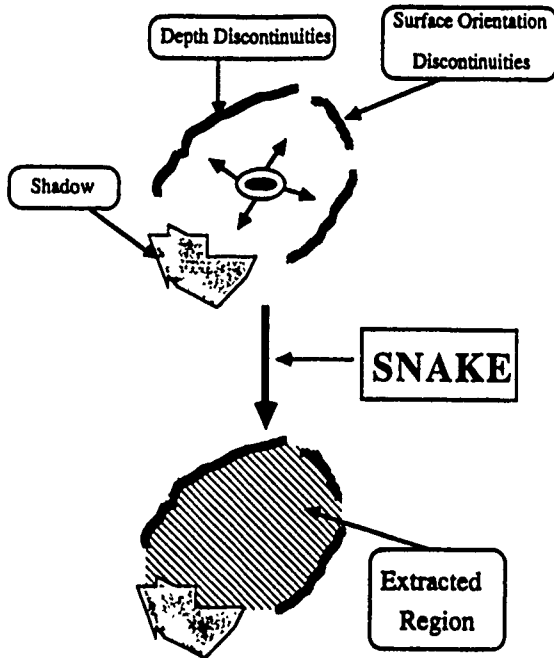


(b)

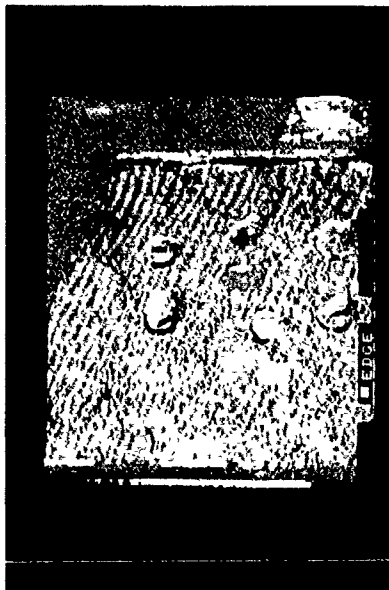


(c)

Figure 1: Rock scene: (a) Input Scene; (b) Range data; (c) Extracted features. The black pixels indicates either discontinuities or shadows.



(a)



(b)

Figure 2: Segmentation: (a) Overview of the segmentation algorithm (b) Segmentation result.

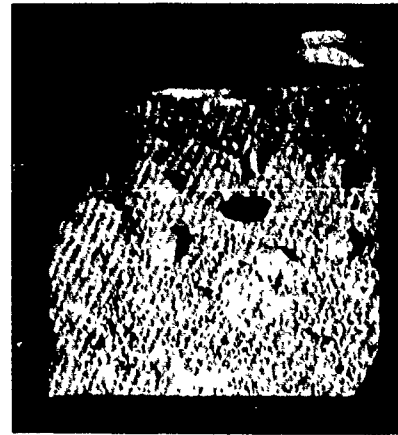


Figure 3: Shape representation by Superquadrics.

Once an object has been represented by a superquadric, the system examines its size parameters to decide whether the rock is small enough to be picked up by the gripper.

If so, the grasping strategy is set up so that the gripper orientation direction is aligned with the rock axis direction, and the approach direction is along the z axis and goes through the mass center of the rock. Those three conditions are sufficient because our gripper has four degrees of freedom. See Figure 4(a).

This configuration yields the minimum potential field given by the relationship between the gripper and the rock represented by the superquadric. Figure 4(b) shows the configuration selected by the system and Figure 4(c) shows the actual grasping.

3 Bin Picking System

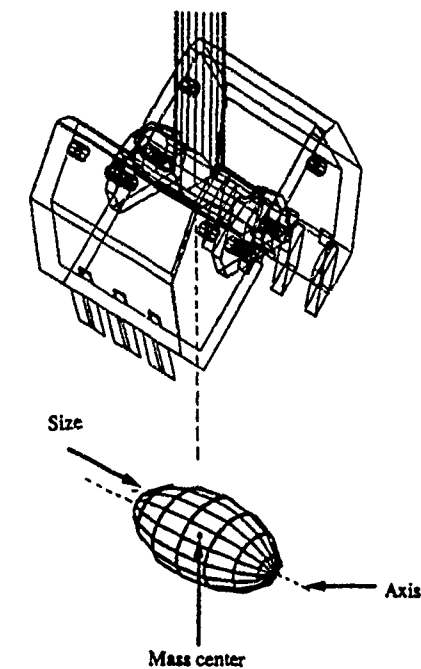
We have developed a bin-picking system based on a vision algorithm compiler for object localization [11, 13]. The central issue in the research was how to build the compiler, which automatically converts a geometric and sensor model into a localization program. In this section, however, we will emphasize the architecture of the run-time system rather than the compiling techniques.

Image Acquisition Sensors Figure 5(a) shows the input scene. The needle image is acquired using a photometric stereo algorithm [12], and the edges are obtained using Canny edge operator [4]. See Figure 5(b).

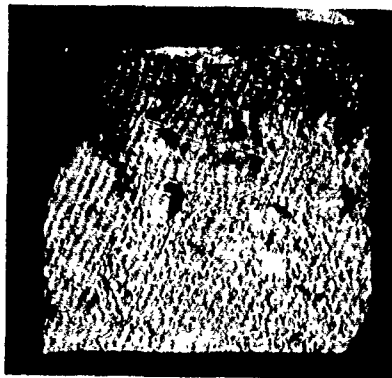
Segmentation From a needle image, two types of features are extracted:

- shadows:

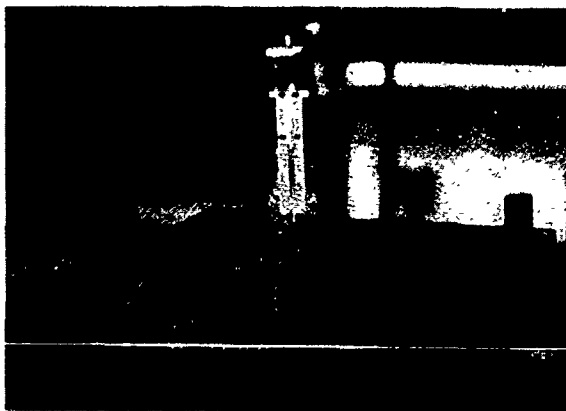
A photometric stereo system projects three lights onto scene; each light generates a shadow region around an industrial part. Since we distribute three lights in a triangle shape, the part located at the top of the bin is surrounded by shadow regions.



(a)



(b)



(c)

Figure 4: Grasping strategy; (a) Grasp plan and a superquadric; (b) a grasp plan; (c) Execution of the plan.

- orientation discontinuities:

From a geometric model of an industrial part, we can determine the angle between two adjacent faces. We can find the minimum angle among these adjacent angles, and use this angle as the threshold for determining orientation discontinuities.

This segmentation method is quite stable due to the characteristics of the scene (bin of industrial parts). Thus, we do not need any detailed segmentation method such as the snake segmentation method in the previous system. Figure 5(c) shows the segmentation results given by simply overlapping shadows and orientation discontinuities; the parts (dotted regions) are separated from each other by shadows and orientation discontinuities (white regions).

Representation Once a region corresponding to an industrial part has been extracted by the segmentation algorithm, several geometric features such as area, inertia and distance between two adjacent regions, are extracted by the localization program. The program performs the localization of an industrial part in two steps: aspect classification and configuration determination. Here, aspects are topologically equivalent classes of appearances of an industrial part. At first, the program classifies one region into one of the aspects, that is, it determines approximate attitude of the parts, corresponding to the aspect. It then uses more precise features such as edges to compute the exact location of the object.

Using the resulting configuration, the program generates an object representation using the geometric model. It also represents neighboring regions by dodecahedral prisms as shown in Figure 6. These dodecahedral prisms are used for constructing a grasping strategy.

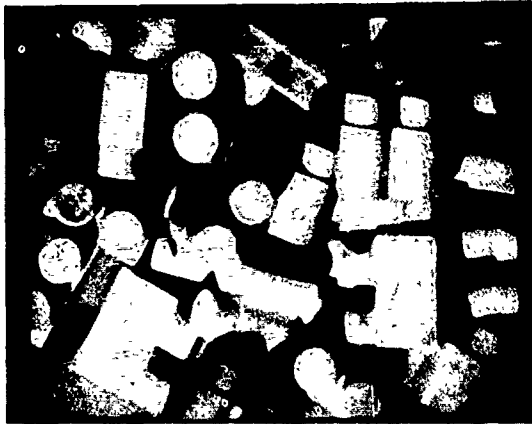
Grasping strategy The grasp configuration should satisfy the following two conditions:

- It should produce a mechanically stable grasp, given the gripper's shape and the part's shape. Such configurations will be called *legal grasp configuration*.
- The configuration must be achieved without collisions with other parts. Grasp configurations are limited by the relationship between the shape of the gripper and the shapes of neighboring obstacles. Such configurations will be called *collision-free grasp configuration*.

In compile mode, possible legal grasp configurations are compiled and stored at each aspect in a grasp catalogue as shown in Figure 7(a).

In execution mode, the legal grasp configurations for the computed aspect are retrieved from the grasp catalogue. These configurations are then converted into the world coordinate system based on the observed configuration of the industrial part.

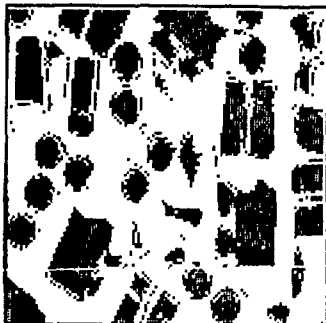
The system has to find a collision-free grasp configuration among these configurations. It generates a cube representation corresponding to the work-space of each legal grasp configuration in the geometric representation. Then, the system examines whether the intersection exists between the gripper work space cube and the obstacle prisms in the geometric representation.



(a)



(b)



(c)

Figure 5: Bin of parts: (a) Input scene; (b) Needle map and edge map; (c) Segmentation result [11].

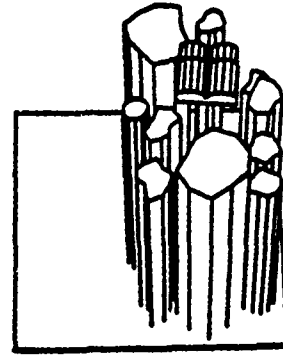


Figure 6: Recognition result [11].

Among the possible collision free configurations found by the system, the optimal configuration (currently the one nearest to the part's mass center) is chosen, as shown in Figure 7(b). The system picks up the industrial part using the configuration as shown in Figure 7(c).

4 Design Analysis

This section analyzes the design process of the two vision systems to illustrate the task-oriented approach.

4.1 Rock-Sampling System

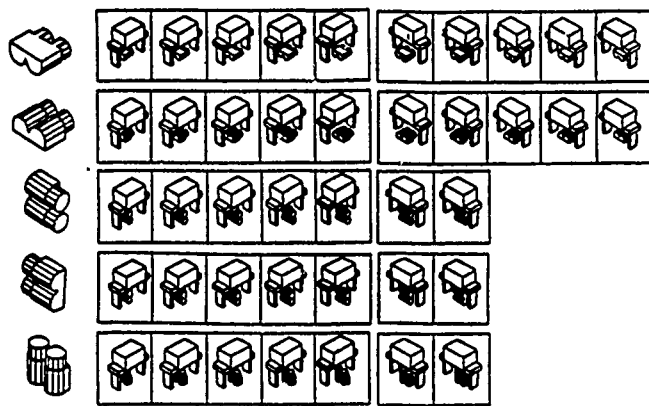
Figure 8 shows the design flow of the rock sampling (RS) system. The design uses the image acquisition method and starts with the below task specifications of the rock sampling.

Task specification The task of this system is to grasp a rock in the sand under the following conditions:

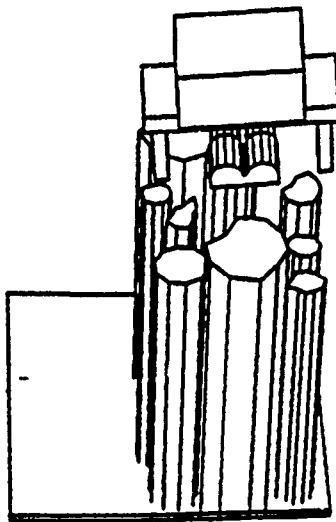
- The rocks are far enough away from each other. No collision occurs between the gripper and the neighboring rocks.
- We can allow the collision between the gripper and the neighboring sand. This is because
 - damaging the neighboring sand grains is not important,
 - the collision between the gripper and neighboring sand does not cause the configuration of the rock to change.
- we do not know the exact shape of a rock beforehand.

Grasping Under this task specifications, we decided to use a spherical grasping. See Figure 10(a). This grasping has the characteristics that

- it requires a large empty volume around an object to be grasped, because all the fingers approach the object from all directions.
- it may grasp the neighboring materials of the object, if any, as well as the object,



(a)



(b)



(c)

Figure 7: Grasp plan; (a) Grasp catalogue, a collection of legal grasp configurations computed beforehand; (b) A collision free configuration; (c) Execution of the grasp plan [11].

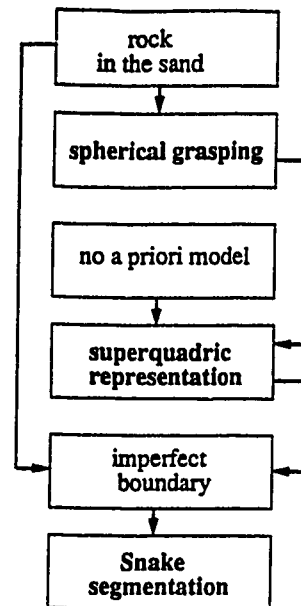


Figure 8: Design flow of Rock-sampling System

- it does not require the precise attitude and position of the object.

In order to realize this spherical grasping, we built a clam-shell gripper as shown in Figure 4(a).

Representation Using a clam-shell gripper imposes two constraints on the representation:

- the expected mass center of a rock should be inside of the gripper
- the expected size of a rock should be smaller than the inner hull of the gripper

While working from only a partial observation of a rock and without any prior knowledge of the rock shape, we still need to recover the above information. We do not need to recover a precise shape representation of a rock, however. For this purpose, we can choose the superquadric representation, because it is described by a few parameters which can be recovered by using a fitting method such as the gradient descent method.

Segmentation For a rock partially buried in the sand, orientation discontinuities and depth discontinuities are small. Thus, it is usually difficult to detect these discontinuities reliably and extract a closed boundary based on them.

Because there is no a priori rock model available, the segmentation cannot be guided by a model as in the bin-picking system. The only available information is that a rock forms a closed boundary. Along this closed boundary, the following three boundary elements exist:

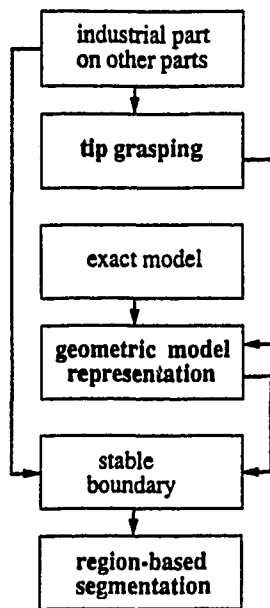


Figure 9: Design flow of Bin-picking System

- shadow boundaries
- orientation discontinuities
- depth discontinuities

Thus, we need a segmentation method which connects these boundary elements and extracts a closed boundary. For this purpose, we employ a model-based segmentation method based on the snake algorithm described in Section 2.

Image Sensor Since this task does not require the rock's precise configuration, the geometric information provided by the range finder is sufficient for carrying out the task. Therefore, we use only a range sensor.

4.2 Bin-picking system

Figure 9 shows the design flow of the bin-picking (BP) system. The design flow starts with the task specification of the bin-picking and specifies all the components down to the image acquisition method.

Task-specification The task of this system is to grasp the top-most industrial part in a bin of parts under the following conditions:

- The parts are close to each other. Some collisions may occur between the gripper and the neighboring parts if we choose a random grasping strategy.
- We should avoid the collision between the gripper and the neighboring parts. This is because

- the collision may cause damage to the neighboring parts,
- it may cause configuration change of the part to be grasped, because the part is supported by the neighboring parts, and thus, we may fail to grasp it.

- We know the exact shape of a part beforehand.

Grasping Under these task-specifications, we decided to use a tip grasping. See Figure 10(f). This grasping has the characteristics that

- it requires only a small volume around an object to be grasped compared to other grasping strategies because only two fingers approach the object from two opposite directions
- it grasps only the object
- it requires the precise attitude and position of the object, because grasping occurs as the contact of two fingers at the same time

In order to realize this tip grasping, we built a parallel-jaw gripper as shown in Figure 7(b).

Representation In order to grasp a part using the parallel-jaw gripper,

- the precise position of two parallel planes should be known.

This constraint implies a polyhedral representation of the object. A sensor typically gives a partial observation of an object. A pair of parallel planes has two opposite surface normals. If one plane is visible from the sensor, it is likely that the other plane is self-occluded from the sensor. Even though the two planes are visible, it is necessary to have an n^2 search out of n observed planes. Thus, we decided not to find such parallel plane pairs at run time.

Instead, we decided to represent a part by a polygonal approximation given by a CAD model, to search such plane pairs in the representation at compile time, and to make the relationship between such pairs and observed part attitude. At run time, we concentrate on recovering the attitude of the part, and recovering the pairs attitude using this relationship.

Segmentation Around the object, we can observe the distinct depth discontinuities, because an industrial part sits on other parts, as opposed to the RS case in which a rock may be partially buried in the sand. Also from the geometric model of the object, we can determine the threshold value used to find surface discontinuities from the minimum angle between adjacent faces.

We use the following facts for segmentation:

- An object boundary is surrounded by a shadow. Since we use a photometric stereo system with three light sources, the object at the top of the bin is always surrounded by a shadow.
- We can predict the threshold value that defines the surface discontinuities by computing angle differences of every face pairs in the model.

Since these two boundaries are distinct and connected, we do not need a method, such as the snake-based segmentation used in RS, to connect them.

Image Sensors This task requires determining the precise attitude of an industrial part of the tip grasping. Unfortunately, region information given by a rangefinder or the photometric stereo is not sufficient for deriving such reliable information. Thus, we have to employ an edge detector as the second sensor so that we can use a fitting method between predicted edges and observed edges given by the edge detector.

5 Towards Systematic Design of Grasping Systems

From examples in the previous sections, we can conclude that for grasping tasks, the following five relationships are the key decisions for designing a vision system:

- task specification and grasping strategy
- grasping strategy and representation
- representation and features
- task specification and segmentation method
- grasping strategy and sensing method

Grasping In order to examine the grasping system design, we have to determine how many grasping strategies are available. Taylor and Schwarz [21] classified the grasping strategies into the following six categories:

- *Spherical grasping* – grasps an object by closing all finger from all directions. The contact occurs at several points on the whole surface of the object so that very stable grasping can be achieved. See Figure 10(a).
- *Cylindrical grasping* – grasps a cylindrical object from all directions in one plane. The contact occurs at the points along the cross-sectional circle. See Figure 10(b).
- *Hook grasping* – pulls an object toward particular directions. The contact occurs at the points along the cross-sectional semicircle. See Figure 10(c).
- *Lateral grasping* – pushes an object on a soft side surface of one finger by the other finger. The contact occurs at a point and points on a plane. See Figure 10(d).
- *Palmar grasping* – grasps the end of bar by closing three fingers. The contact occurs at the three points. See Figure 10(e).
- *Tip grasping* – grasps an object by closing two fingers from two opposite directions. We can achieve very fine grasping. The contact occurs at the two opposite points. See Figure 10(f).

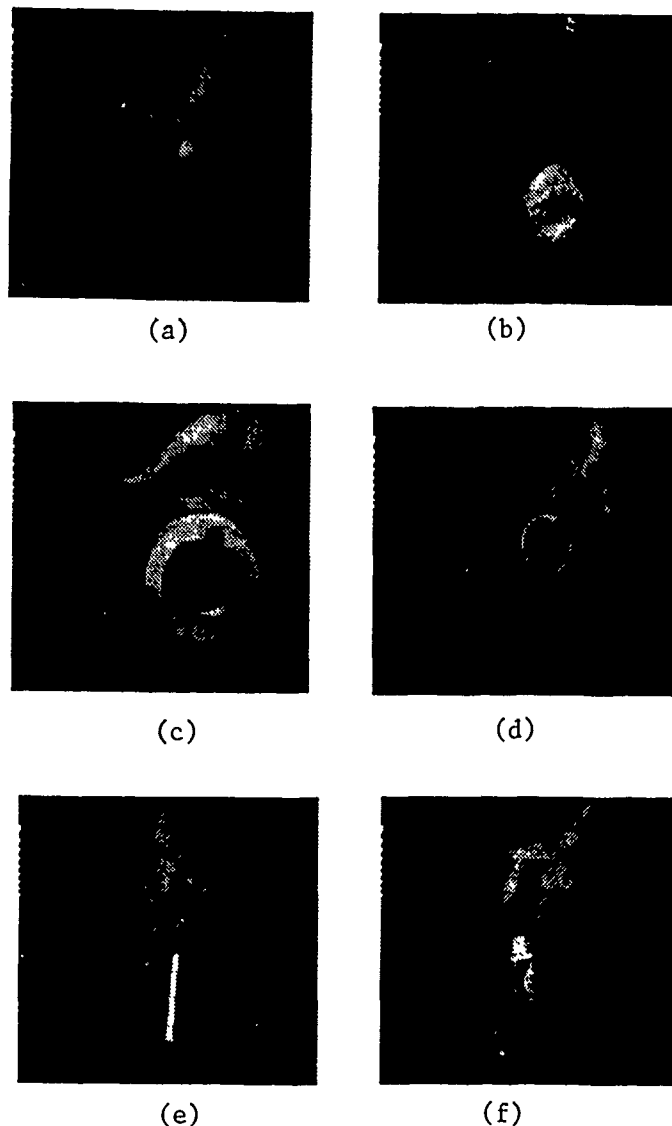


Figure 10: Six basic grasping patterns: (a) Spherical grasping; (b) Cylindrical grasping; (c) Hook grasping; (d) Lateral grasping; (e) Palmar grasping; (f) Tip grasping.

We can realize one of the six patterns of grasping using various kinds of grippers. For example, our parallel jaw gripper can perform the tip grasping and lateral grasping; our clam-shell gripper can achieve the spherical grasping.

The workspace, that is the finger sweeping volume, for the six grasping strategies decreases in accordance with the order listed above: the spherical grasping requires the largest workspace, while the tip grasping requires the smallest workspace. The stability of grasping decreases in the order listed: the spherical grasping has the maximum reliability in grasping, while the tip grasping has the minimum stability in grasping. Thus, we prefer to use the grasping strategies in the listed order, if possible. Thus, assuming that all other considerations are equivalent, our preferences for using a particular grasping strategy runs from those described higher on the list to those covered lower on the list.

Type	Parameters	Representation
Spherical	(a) center (a) radius	superquadrics
Cylindrical	(a) center of cross-section (a) axis direction (a) radius	generalized cylinder superquadrics
Hook	(a) center of cross-section (a) axis direction (a) radius (a) pulling direction	superquadrics plus pulling direction
Lateral	(e) orientation of two planes (e) position of two planes (e) opening distance	two parallel planes (geometric model)
Palmar	(e) center of cross-section (e) radius	cross-sectional shape (geometric model)
Tip	(e) position of points (e) surface orientations	two contact points (geometric model)

Table 1: Possible grasping strategies: (a) and (e) denote approximate and exact information, respectively.

Representation Table 1 summarizes the required functional parameters for a representation by these grasping strategies.

Note that the cylindrical grasping does not require the length of the cylinder. One convenient representation for this grasping strategy is the superquadric.

The hook grasping requires a feature indicating the pulling direction as well as the superquadric representation.

The lateral grasping requires the exact position of the two planes. For this, we need an exact model of an object which is represented using polygons. Since it is difficult to compute the distance between the two planes from an image, we need to have some strong model such as a geometric model of an object and to localize the object in the same way as in the BP system.

In the same way as for the lateral grasping, the palmar grasping requires the knowledge of the exact position of the cross-section, while the tip grasping requires the knowledge of the exact position of two contact points and their orientation. Since it is difficult to extract such information precisely in run time, we need a polygonal approximation such as the one provided by a geometric model of the object.

Features If a geometric model is available, the third issue can be related to the concerns of the automatic feature selection. In this area, Goad proposes a method for selecting appropriate edges for object localization [7]. Bolles and Cain propose a focus feature method which selects important features and less important secondary features [3]. We are developing a method for choosing optimal features for object localization [10]. Other representative methods include [9, 8, 5]. These methods, however, require a strong representation, such as a geometric model of the object, to be handled beforehand.

On the other hand, for weak models such as superquadrics, the third issue can be considered as the problem of finding a suitable surface fitting method. We can use a method such as the gradient descent employed in the RS system in other weak model recovery problems.

Segmentation and Sensors The fourth and fifth issues can be viewed as the problem of determining sensing strategies: what kind of sensor should be used, where it should be located to detect the necessary features, and what kind of features must be extracted.

We have developed tools to model sensors detectability [14] and implemented them into a geometric modeler [15]. We can use this tool to determine the sensing strategies: what kind of boundary elements are obtained, and where they are obtained.

Cowan and Bergman, and Haralick and Shapiro also developed methods to determine the optimal position of the camera and the light source for manipulation tasks [6, 22]. These methods are, however, at the primitive stages. We have to explore more advanced techniques for the systematic design of sensing strategies.

6 Task Oriented Approach

Currently, the majority of vision community agrees with Marr's approach in designing a vision system. Researchers in Marr's school try to design a general purpose vision machine which performs all vision tasks using the single architecture.

Figure 11 shows the paradigm of Marr's approach [17]. From several 2D image clues such as shading, texture, and motion, an intermediate representation ($2 - \frac{1}{2}$ representation) is generated. This representation is based on the viewer-centered coordinate system (the coordinate system attached to an image plane). Then, from this $2 - \frac{1}{2}$ D representation, a final 3D representation, based on the object-centered coordinate system, is generated.

We claim that, depending on the vision task, we should systematically change the architecture of a vision system. We will refer to this as a task-oriented approach; figure 12 shows the paradigm. The basic collection of modules are the same as Marr's except for the existence of a box, labelled as TASK. One particular task for a vision system should govern the choice of representations, vision modules, and image acquisition sensors.

In particular, in order to design a vision system along the task-oriented approach, we propose to consider the following issues:

- task specifications
- required functional capabilities by the task
- representations having such functional capabilities
- features appropriate for extracting such representations
- segmentation methods appropriate for extracting such features and representations
- image sensors appropriate for obtaining such segmentation methods and features

The key element in the task oriented approach is the logical order in which the vision components are selected and built. The approach may be considered as a depth-first approach in that at each level the component most consistent with the previous level is selected. The selection starts from the task specification

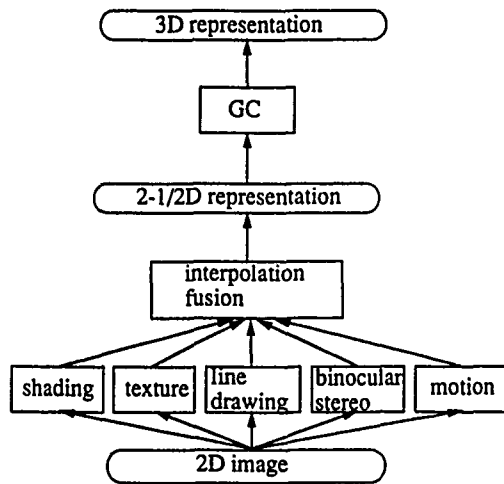


Figure 11: Marr's paradigm

and proceeds all the way down to the sensor selection. This is in contrast with Marr's breadth-first approach in which vision components are developed at each level and are developed independently of the other levels and of the task specification. From the previous examples we can see that this task-oriented approach is critical in building a working system. For example, the BP system would not work if a superquadric representation were used: The difference between the superquadric surface and the actual surface may cause the contact of one finger to occur before the other, thus causing the object to move, and even possibly to fall from the top of the bin.

Toward this systematic selection of components, we have to analyze the role of vision in a task, establish a theory of task analysis and build a vision algorithm compiler. The vision compiler automatically selects appropriate components from a task specification.

7 Conclusion

This paper presented a task-oriented vision approach to the design of vision systems. The task-oriented vision approach proposes to change the architecture of a vision system in a systematic fashion which depends on each task specification. We have presented a task oriented approach for systems that involve the localization and grasping of 3-D objects. In this case, the general methodology involves analyzing the task specification to derive the constraints on and requirements of the vision components. This starts with the type of representation, derived from the type of grasping selected, and continues down to the type of sensor. The task oriented approach is applicable to a wide range of vision systems. The task oriented approach will not only build

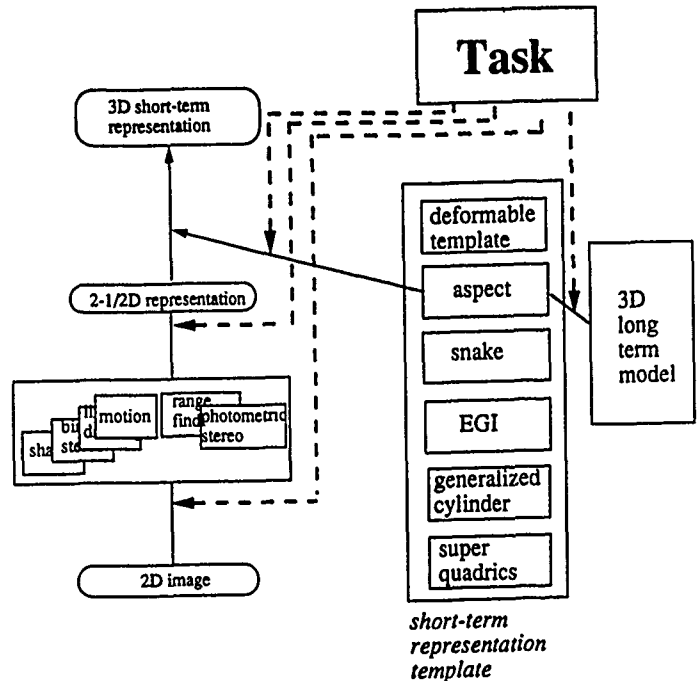


Figure 12: Relationship between task-oriented and bottom-up approaches. The dashed lines indicate the points at which the task description guides the selection of components.

more robust systems but will also give a new direction to vision research.

8 Acknowledgement

The authors wish to thank H. Delingette, T. Choi and Y. Yen for building some of the RS softwares, K.S. Hong and K.D. Gremban for building some of the BP softwares. Takeo Kanade provided many useful comments and encouragements. Kathryn Porsche and Fredric Solomon proofread drafts of this paper and provided many useful comments which have improved the readability of this paper. The authors also thank Shree Nayar, Hideichi Sato, Yoshimasa Fujiwara and the members of the Intelligence Modelling Laboratory, the Robotics Institute, Carnegie Mellon University, for their valuable comments and suggestions.

References

- [1] R. Bajcsy and F. Solina. Three dimensional-object representation revisited. Technical Report MS-CIS-87-19, University of Pennsylvania, Department of Computer and Information Science, March 1987.
- [2] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker. An autonomous rover for exploring mars. *IEEE Computer*, (6), June 1989.
- [3] R. Bolles and R. A. Cain. Recognizing and locating partially visible objects: the local-feature-focus method. *The International Journal on Robotics Research*, 1(3):57-82, 1982.

- [4] J.F. Canny. Finding edges and lines in images. Technical Report AI-TR-720, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1983.
- [5] C.H. Chen and A.C. Kak. A robot vision system for recognizing 3-d objects in low-order polynomial time. *IEEE Trans. on Systems, Man, and Cybernetics*, 19(6):1535-1563, November/December 1989.
- [6] C.K. Cowan and A. Bergman. Determining the camera and light source location for visual task. In *IEEE Intern Conf Robotics and Automation*, pages 509-514, 1989.
- [7] C. Goad. Special purpose automatic programming for 3D model-based vision. In *Proc. of DARPA Image Understanding Workshop*, pages 94-104. DARPA, 1983.
- [8] C. Hansen and T. Henderson. CAGD-based computer vision. In *Proc. IEEE Computer Society Workshop on Computer Vision*, pages 100-105, Miami Beach, FL, December 1987. IEEE Computer Society.
- [9] M. Hebert and T. Kanade. The 3D profile method for object recognition. In *CVPR*, pages 458-463, San Francisco, June 1985. IEEE computer society.
- [10] K.S. Hong, K. Ikeuchi, and K.D. Gremban. Minimum cost aspect classification: a module of a vision algorithm compiler. In *10th Intern. Conf. on Pattern Recognition*, Atlantic City, N.J., June 1990. (a slightly longer version is available as CMU-CS-90-124).
- [11] K. Ikeuchi. Precompiling a geometrical model into an interpretation tree for object recognition in bin-picking tasks. In *Proc. DARPA Image Understanding Workshop*, Los Angeles, CA, February 1986.
- [12] K. Ikeuchi. Determining a depth map using a dual photometric stereo. *The International Journal of Robotics Research*, 6(1):15-31, 1987.
- [13] K. Ikeuchi and K. S. Hong. Determining linear shape change: Toward automatic generation of object recognition program. Technical Report CMU-CS-88-188, Carnegie Mellon University, Computer Science Department, 1988.
- [14] K. Ikeuchi and T. Kanade. Modeling sensors: Toward automatic generation of object recognition program. *Computer Vision, Graphics, and Image Processing*, (48):50-79, 1989.
- [15] K. Ikeuchi and J. C. Robert. Modeling sensor detectability with vantage geometric/sensor modeler. In *Proc. of DARPA Image Understanding Workshop*, pages 721-746. Science Application, Inc., May 1989. (a slightly longer version is available as CMU-CS-89-120).
- [16] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Intern. Journal of Computer Vision*, 2(1):321-331, 1988.
- [17] D. Marr. *Vision*. Freeman, San Francisco, 1982.
- [18] A. P. Pentland. Perceptual organization and the representation of natural form. *Artificial Intelligence*, 28(2):293-331, 1986.
- [19] K. Sato, H. Yamamoto, and S. Inokuchi. Range imaging system utilizing nematic liquid crystal mask. In *International Conf. on Computer Vision*, pages 657-661, London, 1987.
- [20] F. Solina. Shape recovery and segmentation with deformable part model. Technical Report MS-CIS-87-111, Univeristy of Pennsylvania, Department of Computer and Information Science, December 1987.
- [21] C.L. Taylor and R.J. Schwarz. The anatomy and mechanics of the human hand. In *Artificial Limbs*, volume 2, pages 22-35, 1955.
- [22] S. Yi, R.M. Haralick, and L.G. Shapiro. Automatic sensor and light source positioning for machine vision. In *Proceedings of the 10th Intern. Conf. on Pattern Recognition*. IEEE Computer Society, 1990.

Recognition and Positioning of Piecewise Algebraic Objects *

Gabriel Taubin and David B. Cooper

Laboratory For Engineering Man/Machine Systems
Division of Engineering
Brown University
Providence, RI 02912

1 Introduction

In this paper we describe our approach to 3D rigid object recognition and positioning from range data. We will restrict our analysis to piecewise algebraic objects, that is, solid objects whose boundary surfaces can be described as finite collections of smooth surface patches, and each of these patches as a subset of an algebraic surface, the set of zeros of a polynomial in three variables. We will also show that the same techniques apply to the recognition and positioning of rigid objects from sparse nonplanar curve data, and to the recognition and positioning of rigid two-dimensional contours from edge maps.

The paper introduces new ideas for computationally attractive fitting of 3D surfaces, 2D curves, and 3D curves to data, 2D and 3D object representation, and geometric invariants. These geometric invariants appear to be very important for dealing with the situation of a large number of different possible objects in the presence of occlusion and clutter.

An implicit surface is the set of zeros of a smooth function $f: \mathbb{R}^3 \rightarrow \mathbb{R}$ of three variables

$$Z(f) = \{(x_1, x_2, x_3) : f(x_1, x_2, x_3) = 0\}.$$

For example, figure 1 shows an implicit surface which is the set of zeros of the third degree polynomial $f(x_1, x_2, x_3) = x_1^2 + x_2^2 - x_3(x_3^2 - 1) - 1$.

Similarly, an implicit 2D curve is the set of zeros of a smooth function $f: \mathbb{R}^2 \rightarrow \mathbb{R}$ of two variables

$$Z(f) = \{(x_1, x_2) : f(x_1, x_2) = 0\},$$

and an implicit 3D curve is the intersection of two surfaces, the set of zeros of a two dimensional vector function $\mathbf{f}: \mathbb{R}^3 \rightarrow \mathbb{R}^2$ of three variables

$$Z(\mathbf{f}) = \{(x_1, x_2, x_3) : \mathbf{f}(x_1, x_2, x_3) = 0\}.$$

The representation of curves and surfaces in implicit form, as opposed to parametric form, has many advantages. In the first place, an implicit curve or surface maintains its implicit form after a change of coordinates, that is, if a set of points can be represented as a subset of an implicit curve or surface in one coordinate system, so

*This research was partially supported by an IBM Fellowship, NSF Grant IRI-8715774, and NSF-DARPA Grant IRI-8905436

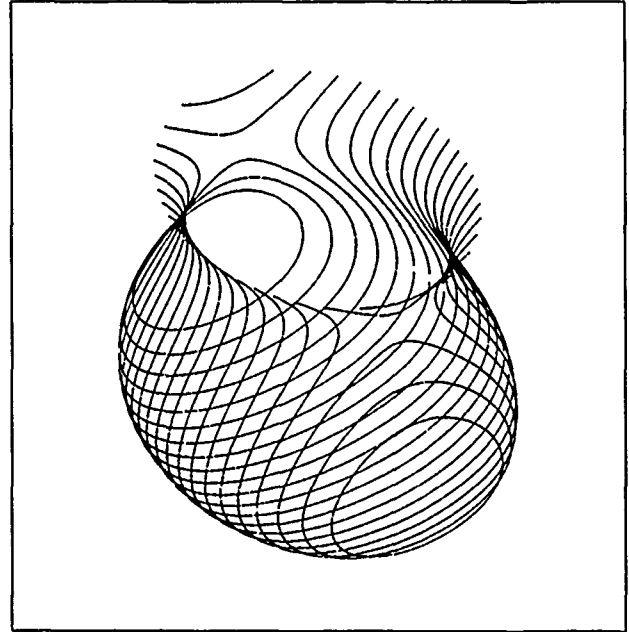


Figure 1: Implicit surface defined by the third degree polynomial $f(x_1, x_2, x_3) = x_1^2 + x_2^2 - x_3(x_3^2 - 1) - 1$.

can it be in any other coordinate system. That is not the case with data sets represented as graphs of functions of two variables, i.e., as depth maps, the patch descriptors produced by many well known segmentation algorithms. In the second place, the union of two or more implicit curves or surfaces can be represented as a single implicit curve or surface, the set of zeros of the product of the functions which define the individual curves or surfaces

$$Z(f_1) \cup Z(f_2) \cup \dots \cup Z(f_n) = Z(f_1 \cdot f_2 \cdot \dots \cdot f_n),$$

so that groups of curve or surface patches, or eventually a whole object, can be represented as a subset of a single implicit curve or surface. For example, the union of two cylinders

$$\{x : x_1^2 + (x_3 - 1)^2 - 4 = 0\} \cup \{x : x_2^2 + (x_3 + 1)^2 - 4 = 0\}$$

shown in figure 2, is the surface defined by the set of zeros of the product

$$\{x : (x_1^2 + (x_3 - 1)^2 - 4)(x_2^2 + (x_3 + 1)^2 - 4) = 0\}. \quad (1)$$

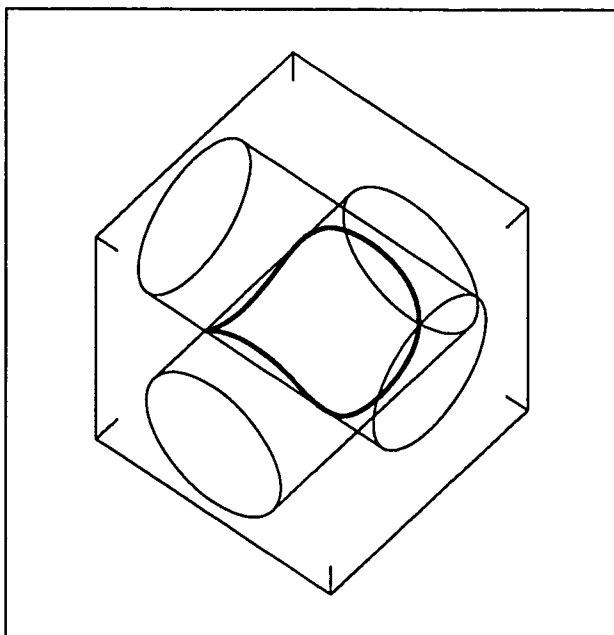


Figure 2: A pair of cylinders represented as a single fourth degree surface.

Hence, a single fourth degree polynomial can represent a pair of cylinders, and this is true for arbitrary cylinders, e.g., a pair that do not intersect. This property relaxes the requirements on a segmentation algorithm, and it is very important in regard to the matching problem, allowing the matching of groups of patches at once.

Towards building a recognition and positioning system based on implicit curves and surfaces, we have to address the following topics:

1. How to efficiently fit implicit curves and surfaces to data, and how to segment a data set into meaningful regions, each consisting of an implicit curve or surface patch.
2. How to match implicit curves and surfaces.
3. Given a pair of matching curves or surfaces, where one is an Euclidean transformation of the other, how to recover the transformation which transforms the first curve or surface into the second one.
4. How to deal with object recognition and position estimation when occlusion is prevalent.
5. How to organize and index into a data base of known objects for purposes of object recognition and position estimation.

The first problem we have to deal with is how to fit implicit curves and surfaces to data. Given a family of implicit functions parameterized by a finite number of parameters, and a finite set of points in space, assumed to belong to the same curve or surface, we would like to fit an implicit curve or surface to the data by estimating the parameters which minimize the mean square distance from the data points to the curve or surface defined by those parameters.

We are primarily interested in two families of implicit functions. One is the *linear model*, where a member of the family is an arbitrary linear combination of a finite number of fixed functions

$$\begin{aligned} f(x) &= F_1 X_1(x) + \dots + F_h X_h(x) \\ &= (F_1, \dots, F_h) \cdot (X_1(x), \dots, X_h(x))^t \\ &= F X(x) \end{aligned}$$

For example, for polynomials of degree ≤ 2 we can take

$$X(x) = (1, x_1, x_2, x_3, x_1^2, x_1 x_2, x_1 x_3, x_2^2, x_2 x_3, x_3^2)^t,$$

the vector of monomials of degree ≤ 2 . Equation (1) is an example of the linear model where $X(x)$ is the vector of monomials of degree ≤ 4 , and F is a row vector of polynomial coefficients. The other family is the *rigid model*, where a member of the family is the composition of a fixed implicit function with an arbitrary Euclidean transformation, that is, the curves or surfaces described by members of this family are all the possible translations and rotations of a fixed curve or surface, and the parameters describe the translation and rotation. Here

$$f(x) = g(T(x))$$

where $Z(g)$ is a fixed curve or surface, and $T(x) = Ax + b$ is an Euclidean transformation, so that the curve or surface $Z(f)$ is equal to the curve or surface $Z(g)$, but after the translation and rotation defined by T^{-1}

$$Z(f) = Z(g \circ T) = T^{-1}[Z(g)].$$

Fitting within the linear model produces unconstrained curve or surface patches, while fitting within the rigid model yields position estimates, the position the original curve or surface has to be moved to in order to minimize the mean square distance to the data. Fitting within the rigid model can be seen as a generalized template matching procedure.

Unfortunately, there is no closed form expression for the distance from a point to a generic implicit curve or surface, not even for algebraic curves or surfaces, and iterative methods are required to compute it. In section 2 we replace the real distance from a point to an implicit curve or surface by a first order approximation. The mean value of this function, on a fixed set of data points, is a smooth nonlinear function of the coefficients, and can be *locally* minimized using well established nonlinear least squares techniques. However, since we are interested in the global minimum, and these numerical techniques find local minima, we still need good initial estimates for the two cases of interest, the linear model and the rigid model.

In order to find a good initial estimate for the linear model, we replace the performance function. Instead of the approximate mean square distance we use a new approximation, turning the difficult multimodal optimization problem into a generalized eigenproblem. The curves or surfaces computed by this *generalized eigenvector fit* method usually produces good fits. The fits are often satisfactory, not requiring further improvement, and the required computation is modest and practical.

The search for initial estimates for the rigid model is based on a theory of Euclidean invariants of algebraic

curves and surfaces, introduced in section 3, because we are dealing with range data. Other researchers have used projective invariants for fitting implicit curves to the projection of 3D curves [Forsyth *et al.*, 1990]. This theory of Euclidean invariants will let us decide whether two curves or surfaces of the same degree match or not. A positive answer to the matching problem would mean that the second curve or surface is almost equal to the first one, but after an unknown Euclidean transformation. If two curves or surfaces match, the theory also lets us recover the Euclidean transformation which transforms the first curve or surface into the second one.

For object recognition, we are developing a system that involves indexing into a data base of objects represented by features consisting of groups of moderately high degree algebraic surfaces. These high degree algebraic surfaces have much more discriminating power than does an individual low degree algebraic surface such as a plane or a quadric surface. Two types of representations are presently under consideration. One is the representation of a collection of a few low degree algebraic surfaces by a single algebraic surface of higher degree. For example, representing three planes by the set of zeros of a single third degree polynomial, the product of the three first degree polynomials, each representing one plane, or a quadric and a cubic surface by a single fifth degree algebraic surface. The simple low degree primitive surfaces used are those that can be found with modest computation. Exact segmentation is not necessary. Partial occlusion is not a problem; a primitive surface can be estimated from a portion of the primitive surface data. Once the primitives are found in the data, groups are then represented by single higher degree algebraic surfaces. The other type of representation are the interest regions, which are spherical regions in which the data is not well represented by a low degree algebraic surface, such as first or second degree, but is well approximated by an algebraic surface of one degree higher. For example, a region occupied by a portion of two intersecting cylinders would be represented exactly by a fourth degree surface and poorly by a lower degree surface if enough of the surfaces were sensed. More generally, a fourth degree surface might capture a chunk of information useful for recognition purposes on a natural irregular surface such as a face, whereas a lower degree surface might not. Useful interest regions are those having the stability that the polynomial does not depend on the exact placement of the sphere specifying the region of data to be used. For this approach, sphere sizes should be chosen such that most spheres will contain data well approximated by low degree surfaces, and only a few will require representation by higher degree surfaces. These higher degree surfaces then contain considerable discriminatory power for object recognition. Figure 3 show an example of an interest region.

In this way we can deal with the occlusion problem. Note that the members of a group of detected patches do not even have to be connected, so that hypotheses of objects and their positions can be generated from more global information, and this procedure can be implemented using a voting scheme, such as a generalized

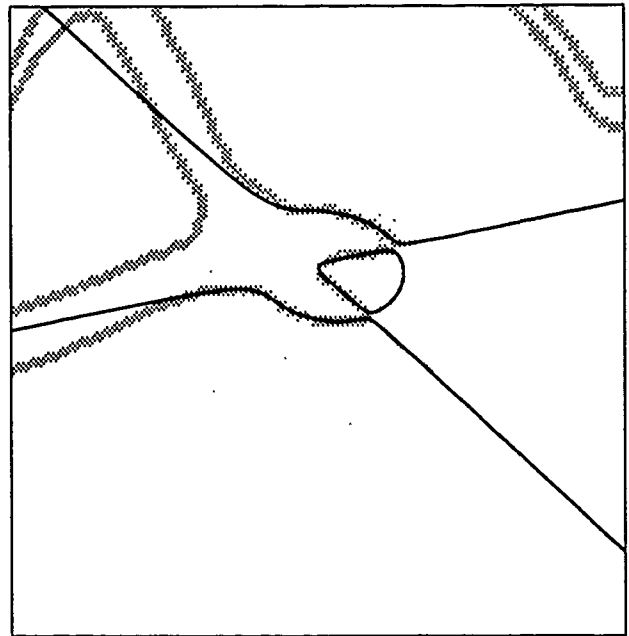


Figure 3: Fourth degree interest region

Hough transform or geometric hashing [Ballard, 1981, Lamdan and Wolfson, 1988, Bolle *et al.*, 1989, Taubin *et al.*, 1989].

2 Implicit Curve and Surface Fitting

Let $\mathcal{D} = \{p_1, \dots, p_q\}$ be a set of n -dimensional data points, and let $Z(\mathbf{f})$ be the set of zeros of $\mathbf{f} = \phi_\alpha : \mathbb{R}^n \rightarrow \mathbb{R}^k$, so that, $Z(\mathbf{f})$ is a 2D curve when $n = 2$ and $k = 1$, it is a surface when $n = 3$ and $k = 1$, and it is a 3D curve when $n = 3$ and $k = 2$. In this section we describe algorithms for fitting an implicit curve or surface $Z(\mathbf{f})$ to the data set \mathcal{D} by approximately minimizing the mean square distance

$$\frac{1}{q} \sum_{i=1}^q \text{dist}(p_i, Z(\mathbf{f}))^2$$

from the data points to the curve or surface.

In general, the distance from a point $x \in \mathbb{R}^n$ to the set of zeros $Z(\mathbf{f})$ cannot be computed by direct methods. The case of a linear map is an exception, in which case the Jacobian matrix $D\mathbf{f}(x)$ is constant, and we have the identity

$$\mathbf{f}(y) \equiv \mathbf{f}(x) + D\mathbf{f}(x) \cdot (y - x).$$

Note that for a surface or 2D curve, $D\mathbf{f}(x) = \nabla \mathbf{f}(x)^t$, and for a 3D curve, $D\mathbf{f}(x)$ is a two row matrix where each row is a transposed gradient vector.

The unique point \hat{y} that minimizes the distance $\|y - x\|$ to x , constrained by $\mathbf{f}(y) = 0$, is given by

$$\hat{y} = x - D^\dagger \mathbf{f}(x),$$

where D^\dagger is the pseudoinverse [Duda and Hart, 1973, Golub and Van Loan, 1983] of $D\mathbf{f}(x)$, so that the square of the distance from x to $Z(\mathbf{f})$ is

$$\text{dist}(x, Z(\mathbf{f}))^2 = \mathbf{f}(x)^t [D\mathbf{f}(x) \cdot D\mathbf{f}(x)^t]^{-1} \mathbf{f}(x).$$

In the general case, where $f(x)$ is not a first degree polynomial, we do not have an identity, but an approximation

$$\text{dist}(x, Z(f))^2 \approx f(x)^t [Df(x) \cdot Df(x)^t]^{-1} f(x). \quad (2)$$

For $k = 1$, the case of a 2D curve or 3D surface, the Jacobian has only one row $Df(x) = \nabla f(x)^t$, and (2) reduces to

$$\text{dist}(x, Z(f))^2 \approx \frac{f(x)^2}{\|\nabla f(x)\|^2}. \quad (3)$$

Note that this distance is the value of the function scaled down by the rate of growth at the point. Due to lack of space, we will continue the development for 2D curves and 3D surfaces, but all the results extend to 3D curves as well [Taubin, 1988a, Taubin, 1988b, Taubin, 1990].

Since we are interested in fitting curves and surfaces to data in a finite number of steps, we will restrict ourselves to families of maps described by a finite number of parameters. Let us choose a smooth function $\phi: \mathbb{R}^{r+n} \rightarrow \mathbb{R}^k$ defined almost everywhere. From now on we will only consider maps $f: \mathbb{R}^n \rightarrow \mathbb{R}$ which can be written as

$$f(x) \equiv \phi(\alpha, x),$$

for certain $\alpha = (\alpha_1, \dots, \alpha_r)^t$, in which case we will also write $f = \phi_\alpha$. We will refer to $\alpha_1, \dots, \alpha_r$ as the *parameters* and to x_1, \dots, x_n as the *variables*. The family of all such maps will be denoted

$$\mathcal{F} = \{f: \exists \alpha f = \phi_\alpha\},$$

we will say that ϕ is the *parameterization* of the family \mathcal{F} .

Now, we will fit curves or surfaces to data points by minimizing the *approximate mean square distance* from the data set \mathcal{D} to the set of zeros of $f = \phi_\alpha$

$$\Delta_D^2(\alpha) = \frac{1}{q} \sum_{i=1}^q \frac{f(p_i)^2}{\|\nabla f(p_i)\|^2}. \quad (4)$$

The problem of computing a *local minimum* of (4) is a nonlinear least squares problem, and it can be solved using several iterative methods [Dennis and Shnabel, 1983], such as the Levenberg-Marquardt algorithm [Levenberg, 1944, Marquardt, 1963, Moré *et al.*, 1980]. Every local minimization algorithm requires a good starting point, and since we are interested in the *global minimization* of (4), when using the Levenberg-Marquardt algorithm we need a method to choose a good initial estimate. In related work, Ponce and Kriegman [Ponce and Kriegman, 1989] have used the Levenberg-Marquardt algorithm for fitting the projections of the occluding boundaries of algebraic surfaces to 2D edges.

Let us consider the linear model first, leaving the rigid model for the next section. In the linear model the maps can be written as

$$f(x) = F_1 X_1(x) + \dots + F_h X_h(x) = FX(x),$$

where $F = (F_1, \dots, F_h)$ is a row vector of coefficients, and $X = (X_1, \dots, X_h)^t: \mathbb{R}^n \rightarrow \mathbb{R}^h$ is a fixed map, and the parameter vector is just $\alpha = F^t$. There exist certain families of implicit curves or surfaces, such as

those which define straight lines, circles, planes, spheres and cylinders, having the value of $\|\nabla f(x)\|^2$ constant on $Z(f)$. In those cases we have

$$\frac{1}{q} \sum_{i=1}^q \frac{f(p_i)^2}{\|\nabla f(p_i)\|^2} \approx \frac{\frac{1}{q} \sum_{i=1}^q f(p_i)^2}{\frac{1}{q} \sum_{i=1}^q \|\nabla f(p_i)\|^2},$$

In the linear model, the right hand side of the previous expression reduces to the quotient of two quadratic functions of the parameters

$$\frac{\frac{1}{q} \sum_{i=1}^q f(p_i)^2}{\frac{1}{q} \sum_{i=1}^q \|\nabla f(p_i)\|^2} = \frac{FMF^t}{FN F^t}, \quad (5)$$

where the matrices M and N , are nonnegative definite, symmetric, and only functions of the data points:

$$M = \frac{1}{q} \sum_{i=1}^q [X(p_i)X(p_i)^t],$$

$$N = \frac{1}{q} \sum_{i=1}^q [DX(p_i)DX(p_i)^t].$$

The new problem, the minimization of (5), reduces to a generalized eigenvalue problem, with the minimizer being the eigenvector corresponding to the minimum eigenvalue of the pencil

$$F(M - \lambda N) = 0.$$

The *generalized eigenvector fit* can be extended to 3D curves as well, where the solution is given by the eigenvectors corresponding to the *two* least eigenvalues. For example, figure 4 shows the result of fitting an implicit 3D curve, defined by the intersection of two general quadric surfaces, to the data points using the generalized eigenvector fit algorithm.

For fitting at modest computational cost, we use the performance function given by the right side of (5), also in the general case where $\|\nabla f(x)\|^2$ is not constant on $Z(f)$.

3 Geometric Matching of Algebraic Curves and Surfaces

Our solution to the algebraic curve and surface matching problem, that is, being able to decide whether two polynomials of the same degree define almost the same curve or surface, but in different positions, and, after a positive answer to recover the transformation, is to define for every algebraic curve or surface an intrinsic frame of reference. By this we mean, a center and an orthonormal basis, functions of the coefficients of the polynomials, but which are rigidly attached to the curves or surfaces that they define. This *intrinsic frame of reference* is commonly referred to as the *object coordinate system* in the Computer Vision literature. In our object coordinate system, the object center is at the origin, and the orthonormal basis for the object coincides with the coordinate unit vectors. After recomputing the coefficients of two polynomials with respect to their intrinsic frames of reference, i.e., their object coordinate systems, we can look at the new coefficients and decide whether

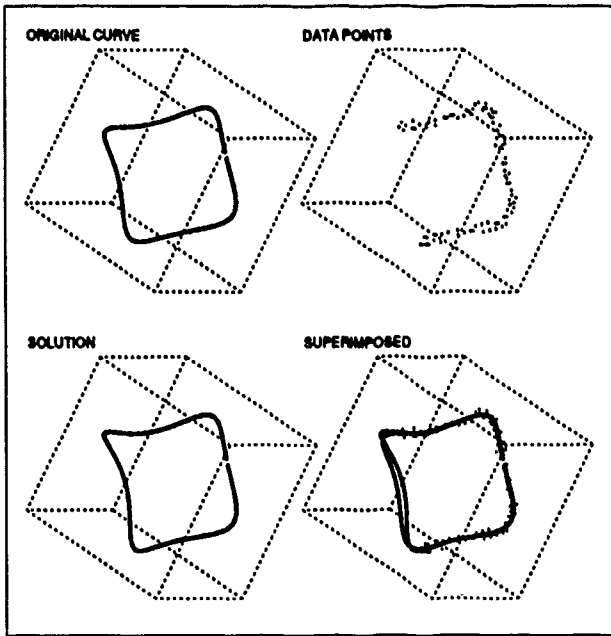


Figure 4: 3D curve fitted to data using the generalized eigenvector fit algorithm

there is a match or not. If the two curves or surfaces are the same, their coefficient vectors should be the same. Then, the rotation and translation which transform the first curve or surface into the second one can be easily computed from the corresponding intrinsic frames of reference. Though we use an intrinsic frame of reference that we have found to be easy to compute, any rotation of these can also be used. The polynomial matching measure that we use has the desirable property of being invariant to such rotation.

If $x' = Ax + b$ is a nonsingular Euclidean transformation, and f is a polynomial, we denote by f' the unique polynomial which satisfies the polynomial identity

$$f(x) \equiv f'(x') ;$$

explicitly, $f'(x') = f(A^{-1}(x' - b))$. If we look at the Euclidean transformation as a change of coordinate system or frame of reference, then both $Z(f')$ and $Z(f)$ describe the same set of points which have different coordinates in the two different coordinate systems. We will define, for every polynomial f , an *intrinsic* frame of reference, that is, a matrix $A = A_f$ and a vector $b = b_f$ which are functions of the coefficients of f such that $Z(f')$ is located in a *canonical position and orientation*. Then if $Z(f')$ and $Z(g')$ are in the canonical positions for the polynomials f and g , respectively, we will say that $Z(f)$ and $Z(g)$ match exactly if $f' \equiv g'$, except for a nonzero multiplicative constant, where $f'(x') = f(A_f^{-1}(x' - b_f))$ and $g'(x') = g(A_g^{-1}(x' - b_g))$.

For example, figures 5 and 7 show two cubic 2D curves given by the union of three straight lines extracted from the edge images, and figures 6 and 8 show the corresponding frames of reference.

This approach also lets us make approximate matches by comparing the vectors of coefficients of the two poly-

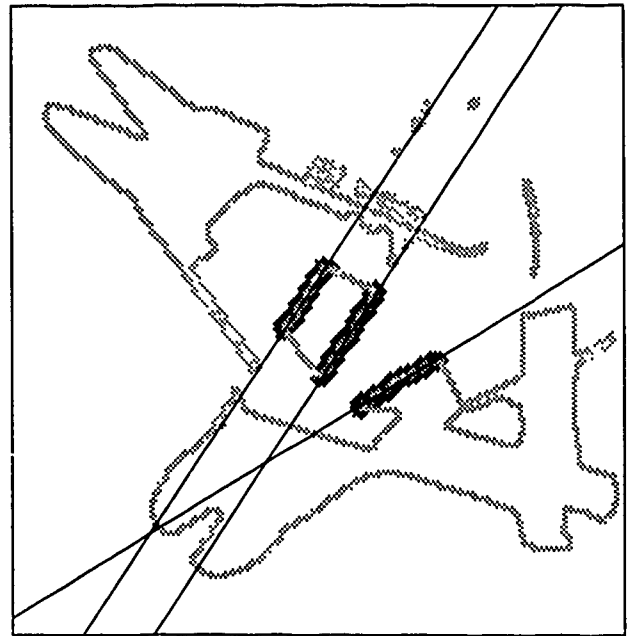


Figure 5: Cubic 2D curve, union of the three straight lines fitted to the data points in the dark region

nomials. Once an inner product $\langle \cdot, \cdot \rangle$ is chosen in the space of polynomials of degree $\leq d$, the following number is a *matching measure*

$$\text{match}(f, g) = \frac{\langle f', g' \rangle^2}{\|f'\|^2 \|g'\|^2},$$

where as usual $\|f\|^2 = \langle f, f \rangle$ is the norm of f with respect to the given inner product. This inner product should include appropriate weightings for the different components of the polynomial coefficient vectors.

We will decompose the computation of the intrinsic frame of reference in two parts. We will first find the translation vector or *center*, and then the rotation matrix or *canonical orientation*, but we need to introduce some nomenclature first.

By a *homogeneous* polynomial ϕ of degree d we mean one that can be written in a unique way as a linear combination of monomials of degree d

$$\phi(x) = \sum_{|\alpha|=d} \frac{1}{\alpha!} T_\alpha x^\alpha,$$

where the vector of nonnegative integers $\alpha = (\alpha_1, \dots, \alpha_n)^t$ is a *multiindex* of size $|\alpha| = \alpha_1 + \dots + \alpha_n$, $\alpha! = \alpha_1! \dots \alpha_n!$ is a multiindex factorial, $\{T_\alpha : |\alpha| = d\}$ is the set of coefficients of ϕ , and $x^\alpha = x_1^{\alpha_1} \dots x_n^{\alpha_n}$ is the *monomial* of degree d associated with the multiindex α .

A homogeneous polynomial can also be written without the multiindex factorial coefficients, but their use provides us with a very important tool. If $\{T_\alpha : |\alpha| = d\}$ and $\{U_\alpha : |\alpha| = d\}$ are the sets of coefficients of the homogeneous polynomials ϕ and ψ of degree d , then the following expression

$$\langle \phi, \psi \rangle = \sum_{|\alpha|=d} \frac{1}{\alpha!} T_\alpha U_\alpha$$

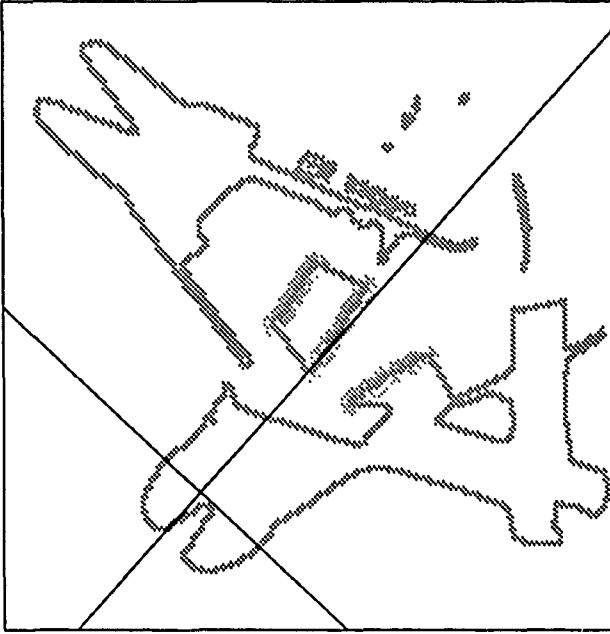


Figure 6: Intrinsic frame of reference for the curve in figure 5

defines an inner product in the vector space of homogeneous polynomials of degree d , which is *invariant* under orthogonal transformations [Helgason, 1984], that is, if $x' = Ax$ is an orthogonal transformation, then

$$\langle \phi, \psi \rangle = \langle \phi', \psi' \rangle$$

Hence, though we chose the orientation of an object within its object reference frame in one way for convenience, it could have been chosen in any other way, and the matching measure we use would remain unchanged.

Our definition of *center* of a 2D curve or 3D surface of degree $d \geq 2$ is a generalization to $d > 2$ of the well known case of a nonsingular quadratic curve or surface. Every polynomial f of degree d can be written in a unique way as

$$f(x) = \sum_{i=0}^d f_i(x),$$

where f_i is an homogeneous polynomial of degree i , and $f_d \neq 0$. For every fixed space vector y , the polynomial $g(x) = f(x+y)$, as a polynomial in x has exactly the same degree d , and so it can also be written in a unique way as a sum

$$f(x+y) = g(x) = \sum_{i=0}^d g_i(x),$$

where the coefficients of the homogeneous polynomials g_i are polynomials in y . Particularly, the term of degree d is invariant

$$g_d \equiv f_d,$$

and the term of degree $d-1$ is given by

$$g_{d-1} \equiv f_{d-1} + y^t \nabla f_d = f_{d-1} + \sum_{i=1}^n y_i \frac{\partial f_d}{\partial x_i}.$$

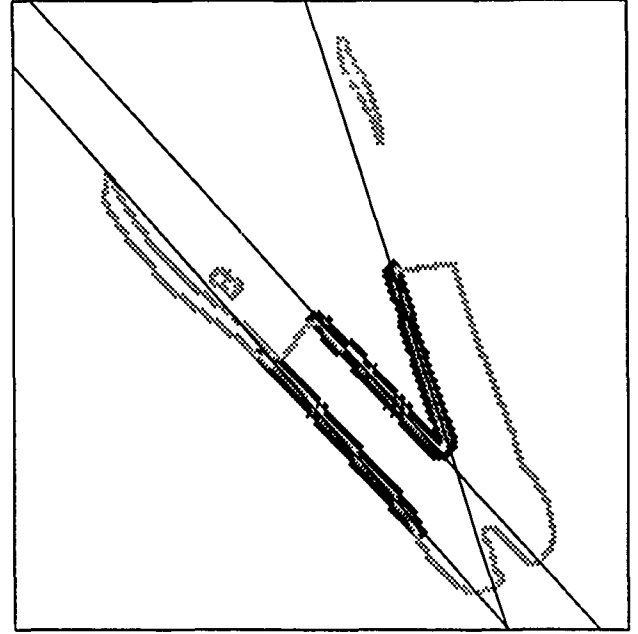


Figure 7: Cubic 2D curve, union of the three straight lines fitted to the data points in the dark region, matching curve in figure 5

We define the *center* of f as the vector y which minimizes the invariant norm of the homogeneous polynomial g_{d-1}

$$\|f_{d-1} + y^t \nabla f_d\|^2,$$

a least squares problem, which has a unique solution if the vectors of coefficients of the partial derivatives of the term of degree d , the homogeneous polynomials $\partial f_d / \partial x_1, \dots, \partial f_d / \partial x_n$, are linearly independent. If they are not linearly independent, then the definition of the center has to be generalized including the terms of lower degree as well, but due to the lack of space we will not cover these cases here.

The *canonical orientation* can be defined in several ways, all of them based on the fact that every symmetric matrix with nonrepeated eigenvalues has an associated set of eigenvalues, thus generating 2^n different orthogonal coordinate systems having unit vectors in the directions of these eigenvectors. For example, given a polynomial f of degree d , which we decompose as a sum of homogeneous polynomials

$$f(x) = \sum_{i=0}^d f_i(x),$$

we consider the symmetric $n \times n$ matrix whose (i, j) -th element is the invariant inner product of the i -th and j -th partial derivatives of f_d with respect to x_i and x_j

$$\left\langle \frac{\partial f_d}{\partial x_i}, \frac{\partial f_d}{\partial x_j} \right\rangle.$$

If this matrix has all different eigenvalues, the *canonical orientation* of f is defined as the orientation induced by the eigenvectors of the matrix. Then, in order to disambiguate among the 2^n different frames of reference, we

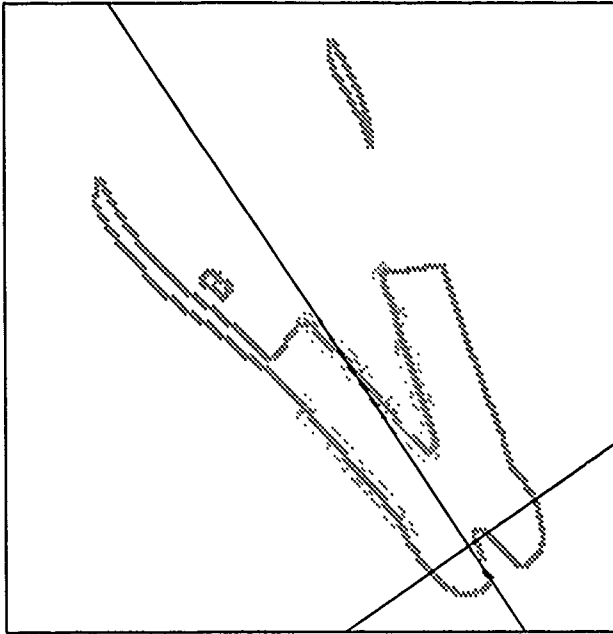


Figure 8: Intrinsic frame of reference for the curve in figure 7

will find the location of certain fixed points, other than the center. Every nonzero component of a fixed point can be used to choose the orientation of the corresponding axis. If the matrix has repeated eigenvalues, we will have to use information provided by the other homogeneous terms of f , as we do for the center, but again, we are not able to cover these special cases here.

We have defined a canonical orientation for a d -th degree polynomial, equivalently, its orientation within its object coordinate system, solely in terms of the coefficients of its d -th degree monomials. Is this a stable representation? This representation should be stable for two reasons. First, the coefficients of the highest degree monomials are significant in the polynomials that we use. They are significant for the interest regions because lower degree polynomials do not fit the data well there. They are significant when each polynomial is a product of a group of low degree polynomials that we use because, again, a single lower degree polynomial would not fit a group of surfaces involved. Second, the regions chosen to be interest regions are those for which the representative polynomials are not sensitive to small changes in the region used. The groups of low degree polynomial surfaces that we use are those that can be easily found, that is, for which the segmentation is very stable, even in the presence of partial occlusion. We have developed other geometric invariants, based on moments for high degree polynomials, but the description of these is too involved for presentation in this paper.

References

- [Ballard, 1981] D.H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111-122, 1981.
- [Bolle et al., 1989] R.M. Bolle, A. Califano, R. Kjeldsen, and R.W. Taylor. Visual recognition using concurrent and layered parameter networks. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 625-631, June 1989.
- [Dennis and Shnabel, 1983] J.E. Dennis and R.B. Shnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, 1983.
- [Duda and Hart, 1973] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.
- [Forsyth et al., 1990] D. Forsyth, J.L. Mundy, A. Zisserman, and C.M. Brown. Projectively invariant representation using implicit algebraic curves. In *Proceedings, First European Conference on Computer Vision*, 1990.
- [Golub and Van Loan, 1983] G. Golub and C.F. Van Loan. *Matrix Computations*. John Hopkins University Press, 1983.
- [Helgason, 1984] S. Helgason. *Groups and Geometric Analysis*. Academic Press, Inc., 1984.
- [Lamdan and Wolfson, 1988] Y. Lamdan and H.J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proceedings, Second International Conference on Computer Vision*, pages 238-249, December 1988.
- [Levenberg, 1944] K. Levenberg. A method for the solution of certain problems in least squares. *Quarterly of Applied Mathematics*, 2:164-168, 1944.
- [Marquardt, 1963] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal of Applied Mathematics*, 11:431-441, 1963.
- [Moré et al., 1980] J.J. Moré, B.S. Garbow, and K.E. Hillstom. User guide for minpack-1. ANL-80-74, Argonne National Laboratories, 1980.
- [Ponce and Kriegman, 1989] J. Ponce and D.J. Kriegman. On recognizing and positioning curved 3D objects from image contours. In *Proceedings, IEEE Workshop on Interpretation of 3D Scenes*, November 1989.
- [Taubin et al., 1989] G. Taubin, R.M. Bolle, and D.B. Cooper. Representing and comparing shapes using shape polynomials. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, June 1989. also IBM Technical Report RC-14292.
- [Taubin, 1988a] G. Taubin. Algebraic nonplanar curve and surface estimation in 3-space with applications to position estimation. Technical Report LEMS-43, Brown University, February 1988. Also, IBM Technical Report RC-13873.
- [Taubin, 1988b] G. Taubin. Nonplanar curve and surface estimation in 3-space. In *Proceedings, IEEE Conference on Robotics and Automation*, May 1988.
- [Taubin, 1990] G. Taubin. Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation. Technical Report LEMS-66, Brown University, January 1990.

Modeling Polyhedra by Constraints

Van-Duc Nguyen and Joseph L. Mundy*

Artificial Intelligence Program
Corporate Research and Development
General Electric Company
Schenectady, NY 12345

Deepak Kapur

Institute for Programming and Logics
Department of Computer Science
State University of New York
Albany, NY 12222

Abstract

A constraint-based modeling system is described. A polyhedral model is represented as a network of nodes and constraints. Nodes are 3D vectors representing the location and orientation of the geometric entities, or measure variables such as length or cosine. Constraints are polynomial equations in the node parameters. Modeling and recognition are viewed as solving for values of the node parameters such that all the constraint equations are satisfied and the mean square error between model and observed shape is minimized.

An approach for solving the constrained minimization problem that emphasizes the elimination of dependent parameters using symbolic algorithms and the best fit between model and observed shape using numerical optimization techniques is described. Buchberger's Gröbner basis algorithm and Ritt-Wu's triangulation algorithm can be used for eliminating dependent parameters as well as for detecting inconsistency among constraints. A modification of the triangulation algorithm is proposed which helps in controlling the growth of the intermediate computations.

1 Introduction

The representation of objects in terms of geometric constraints provides an approach for the integration of many aspects of scene description and recognition. The central idea is that an object is described by a set of symbolic geometric relations which constrain the possible configurations of the entities from which the object is composed. For example, a square is composed of edges which are restricted to equal length and each corner must be a right angle. This description does not specify any specific instance of a square but admits an infinite space of squares according to orientation, location and scale.

In conventional modeling systems, an object is described by incidence relations between faces, edges, and vertices, as well as numerical specification of geometry in terms of vertex positions, face and edge parameters. For polyhedral objects, edges are line segments and faces are bounded planes. For curved objects, the edges are bounded algebraic curves and faces are algebraic surfaces bounded by closed edge contours. The limitation of this conventional representation is that specific geometric relations between model entities are expressed in terms of the numerical parameters associated with vertices, edges and faces.

In order to recover the geometric relations, classification procedures which operate on the numerical data must be established, and tolerances which specify the range of validity of a particular relation must be provided. For example, two lines may be classified as orthogonal if the angle between them is between 89° and 91° . The choice of such tolerances is very difficult and depends on the numerical precision of arithmetic computation.

A major advantage of the constraint description is that the intended relationships between components are explicit. When two lines are intended to be orthogonal, this relationship is directly expressed in the object definition. In order to generate an instance of a constraint-based model the constraint equations must be solved to determine values for the model parameters which simultaneously satisfy all of the equations. It is often the case that the specified equations are fewer than the model parameters, so an additional restriction is imposed so that the model solution agrees as closely as possible with empirical measurements on observable model elements. For example, for the case of the square, we may have measured vertex locations which do not correspond exactly to the corners of an ideal square. It is reasonable to expect that the model parameters maintain the constraints associated with a square and also minimize the mean square error between the model vertices and the measured positions.

This example brings up a general philosophical consideration concerning constraint models which is important to emphasize. In our view, the constraints specified by a user are considered to be exact in the sense of ideal geometric figures. Any error involved in the specification of constraints is absorbed in the errors associated with

*Work at GE was supported in part by the DARPA Strategic Computing Vision Program in conjunction with the Army Engineer Topographic Laboratories under Contract No. DACA76-86-C-0007 and the Air Force Office of Scientific Research under Contract No. F49620-89-C-0033.

empirical observations. To clarify the point, consider the case of house walls which are closely aligned with the direction of gravitational force, through the use of a carpenter level or plumb bob. Now suppose we wish to extract house models from image data. It is reasonable to specify that house walls are exactly aligned with the gravitational field even though a specific house may have some errors in construction. Any error in alignment can be taken up in the error minimization process.

A slight variation of this approach is possible if tolerances on model construction are known in advance. One can specify geometric relations with slack parameters and then establish a tolerance range on these parameters to account for uncertainty in the relation. The specification is still exact in the sense that tolerance bounds are themselves considered to be free of error.

Another major advantage of representation by geometric constraints is that a large range of specific objects can be derived from a single constraint model. The introduction of constraints as the basic representation also permits the direct specification of known relationships between objects and between objects and cameras. The following are several examples of constraint relations which arise in aerial photography:

- Shadow geometry is constrained by sun angle and object boundaries.
- Buildings are in contact with and perpendicular to the terrain surface and usually aligned with roadways.
- Known relations exist between cartographic features as specified by a map.
- Camera viewpoint is often specified by navigational coordinates.

In model-based vision, one can take the general view that recognition corresponds to the existence of a solution for a particular set of model constraints which yields close agreement with the observed image features and relations between image features. In current model-based vision systems, the model structure is usually fixed, and six degrees of freedom associated with model pose are determined to minimize the error between projected model features and image features derived from segmentation. If the agreement is sufficiently close, then recognition is declared for the specific model pose. For more general constraint-based systems, the recognition process is identical except that the object representation usually defines a larger number of degrees of freedom which must be pinned down by the recognition process. The final decision concerning the applicability of a particular constraint depends entirely on the error in predicting observable features. Ultimately, it is necessary to specify error bounds on observations such as vertex position and edge orientation, but the availability of a general constraint representation allows these specifications to be based on a few variances associated with a particular sensor and particular segmentation algorithms.

The idea of representing objects and object configurations in terms of geometric constraints is not recent, but there have been relatively few image understanding systems developed around these concepts. Perhaps

the most ambitious example is the ACRONYM system developed by Brooks [Brooks, 1981]. ACRONYM represented objects in terms of generalized cylinders with constraints expressed in terms of symbolic inequalities. The inequalities provide a direct mechanism for specifying tolerances on the position and orientation associated with the projection of models constructed from generalized cylinder components. Brooks developed an innovative approach to the solution of systems of symbolic inequalities, including orientation relations, but the cost of the algorithm permits only relatively small constraint systems to be solved.

As an interesting historical note, a very early example of constraint modeling is Sutherland's work. In his SKETCHPAD system, Sutherland implemented a constraint solver so that the user could crudely specify drawings by hand [Sutherland, 1963]. The sketches provided the initial guess to an iterative process for solving the constraint system and producing accurate drawings. Another early example is the model-based recognition system developed by Roberts [Roberts, 1965]. His recognition system was based on polyhedral object models which could be mapped into variable shapes by allowing a full projective transformation on the 3D vertex positions. This mapping on 3D space can transform a cube into any rectangular prism and even into a truncated pyramid as in a drawing with vanishing points. Thus a wide variety of shapes can be specified from a fixed polyhedron and a 4x4 homogeneous transformation matrix.

A more recent example, similar to our work, is the development of a complete constraint-based modeling system for mechanical design by Gossard, Light, and Serrano [Light and Gossard, 1982, Serrano and Gossard, 1987]. Their approach is largely numerical using techniques of nonlinear programming. They have observed that large constraint systems can be reliably solved if the Jacobian matrix associated with constraints can be reduced to block diagonal form. Another important observation is that it is easy for users to specify inconsistent constraints and that mechanisms must be developed to automatically detect and isolate such inconsistencies.

The goal of our project is to produce a constraint solver which can handle large constraint systems, perhaps of the order of thousands of equations, and at the same time provide a general constraint representation to permit flexible specification of object descriptions. The focus of our current work is the development of a hybrid system which employs techniques taken from symbolic algebra as well as more conventional numerical optimization algorithms. Our view is that either approach by itself is insufficient to achieve the desired result. Purely numerical approaches may fail to converge or become trapped at a local minimum and it is difficult to detect inconsistent constraint specifications. Purely symbolic approaches become intractable for even small problems due to the exponential growth in the size of algebraic expressions during variable elimination. We believe that there is middle ground where symbolic techniques can be used to compile geometric constraint specifications into efficient numerical modules with well behaved convergence properties. The following sections give details

of our approach and some recent experimental results.

2 Network of Nodes and Constraints

A model is represented by a network of nodes and constraints. Nodes are non overlapping sets of variables. They describe the configurations of component entities such as vertices, edges, faces. Constraints are n-ary relations among nodes. They are represented by equations which must be satisfied by the values of the nodes' variables.

2.1 Nodes as Vectors or Measures

Primary geometric entities in 3D are 0-, 1-, and 2-dimensional manifolds. In the simple case of polyhedral objects, they correspond to vertices, edges, and faces. If we view these entities as sets of points, vertices have 'invariant' locations, whereas edges and faces have invariant orientations which are respectively the edge directions and the face normals. The locations and orientations of the geometric entities can be represented as 3D vectors, with three variables each, corresponding to the three Cartesian coordinates along the x, y, z axes.

In the case of curved objects, the curved edges and faces no longer have invariant orientations. However, their orientations vary slowly and so can be parameterized relative to local coordinate frames centered at their mid points. The parameterized equations usually make explicit salient features such as local symmetries [Faux and Pratt, 1979, Horn, 1986], and so are preferred.¹

To span the spectrum from polyhedral to curved objects, we choose to represent geometric entities by their locations and orientations. This choice reflects the common distinction between linear and angular components in geometry and mechanics. Besides locations and orientations, we also need two measures: length to describe the distance between two locations, and cosine to describe the angle between two orientations. Measures are represented by scalar variables. Coordinate frames [Paul, 1981] and camera projections [Slama, 1980], are represented similarly with measures, location and orientation vectors.

From now on, we restrict ourselves to the case of polyhedral objects. Each vertex, edge, or face has a representative location and orientation, denoted respectively by (\mathbf{v}) , (\mathbf{e}, \mathbf{d}) , or (\mathbf{f}, \mathbf{n}) . The locations of the vertices, edges, and faces are related by linear combinations. The edge orientation is the unit direction vector between its two end vertices. The face orientation is the unit normal vector perpendicular to the first two edge directions from its boundary edges. The hierarchy of geometric entities, figure 1, leads to a similar dependency between the nodes in the network.

2.2 Constraints as Polynomial Equations

All the primary geometric constraints are exhaustively enumerated by finding all pairs between linear and angular entities, table 1. We have location constraint be-

¹Generalized spheres and cones are special cases of curved objects for which it is most efficient to represent with implicit equations, rather than with parameterized equations.

	VERTEX	EDGE	FACE
VERTEX	location	colinearity	coplanarity
EDGE		orientation	coplanarity
FACE			orientation

Table 1: Primary geometric constraints.

tween two locations, orientation constraint between two orientations, and collinearity/coplanarity constraints between the locations and orientations of any two geometric entities with different dimensions.

Polynomials are commonly used in geometric reasoning to describe constraints [Kapur and Mundy, 1989, Ponce and Kriegman, 1989], structures of line drawings [Sugihara, 1986], and geometric invariants [Forsyth *et al.*, In preparation]. Polynomial equations will be used to represent geometric constraints. Polynomial inequations are only needed to describe betweenness constraints.

Location Constraint

Let \mathbf{v}_1 and \mathbf{v}_2 be two vertices, and l be a length measure. The location constraint equalizes the distance between two locations and the length, and is represented by a polynomial equation in second order:

$$((\mathbf{v}_1 - \mathbf{v}_2) \cdot (\mathbf{v}_1 - \mathbf{v}_2)) - l^2 = 0 \quad (1)$$

The first partial derivatives of the location constraint relative to the variables in \mathbf{v}_1 , \mathbf{v}_2 , and l are:

$$[2(\mathbf{v}_1 - \mathbf{v}_2)^T, 2(\mathbf{v}_2 - \mathbf{v}_1)^T, -2l] \quad (2)$$

The location constraint becomes singular with all the first partial derivatives vanishing to zero, as the length l tends towards zero, and the two vertices collapse into one [Nelson, 1985]. To avoid this singularity, we put a nonzero lower bound on l , and treat coincident locations as a special case where the location constraint is described by three first order polynomial equations.

Orientation Constraint

Let \mathbf{d}_1 and \mathbf{d}_2 be the directions of two edges, and α be the angle between the two direction vectors. The orientation constraint equalizes the dot-product of the direction vectors and the cosine of the angle. It is represented by a polynomial equation in second order if we use the cosine as variable, and normalize the magnitude of the direction vectors to 1:

$$\begin{aligned} (\mathbf{d}_1 \cdot \mathbf{d}_2) - \cos \alpha &= 0 \\ (\mathbf{d}_1 \cdot \mathbf{d}_1) - 1 &= 0 \\ (\mathbf{d}_2 \cdot \mathbf{d}_2) - 1 &= 0 \end{aligned} \quad (3)$$

The Jacobian of the above constraints respective to the variables in \mathbf{d}_1 , \mathbf{d}_2 , and α is:

$$\begin{bmatrix} (\mathbf{d}_2)^T & (\mathbf{d}_1)^T & \sin \alpha \\ 2(\mathbf{d}_1)^T & (0)^T & 0 \\ (0)^T & 2(\mathbf{d}_2)^T & 0 \end{bmatrix} \quad (4)$$

The local Jacobian becomes singular with the first row becoming a linear combination of the last two rows, as

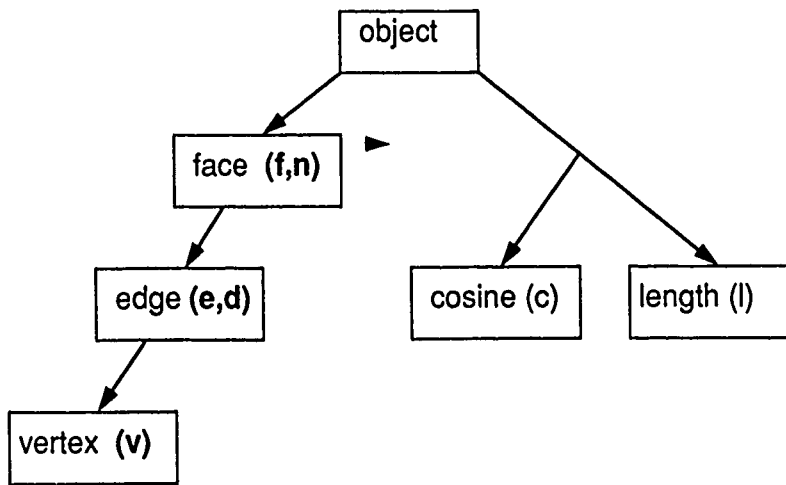


Figure 1: Hierarchy of geometric entities.

α tends to 0 or π , and the two direction vectors become parallel. To avoid this singularity, we put bounds on $\cos \alpha$ and treat parallel orientations as a special case where the constraint is described by three first order equations.

Collinearity and Coplanarity Constraint

1. Collinearity of a vertex to an edge is described by three second order polynomial equations:

$$(\mathbf{v} - \mathbf{e}) \times \mathbf{d} = 0 \quad (5)$$

Only two equations are independent. Since the edge direction can change at run time, we are forced to use all three equations, or we have to express the constraint in the local frame of the edge.

2. Coplanarity of a vertex to a face is described by one second order polynomial equation:

$$(\mathbf{v} - \mathbf{f}) \cdot \mathbf{n} = 0 \quad (6)$$

3. Coplanarity of an edge to a face is described by two second order polynomial equations:

$$\begin{aligned} (\mathbf{e} - \mathbf{f}) \cdot \mathbf{n} &= 0 \\ \mathbf{d} \cdot \mathbf{n} &= 0 \end{aligned} \quad (7)$$

Alignment and attachment between geometric entities are parsed into collinearity or coplanarity constraints. The implicit coplanarity of the vertices and edges belonging to a face are made explicit by a minimum number of independent coplanarity constraints, section 2.3.

Symmetry Constraint

Local symmetry about a point, line, or plane could be generated from operations on vectors and scalars and

the above primary geometric constraints. For example, two vertices v_1 and v_2 are symmetric about the support plane of face f if and only if there exists a point v such that:

$$\begin{aligned} (\mathbf{v} - \mathbf{f}) \cdot \mathbf{n} &= 0 \\ (\mathbf{v}_2 - \mathbf{v}_1) \times \mathbf{n} &= 0 \\ \mathbf{v} - 0.5(\mathbf{v}_2 + \mathbf{v}_1) &= 0 \end{aligned} \quad (8)$$

The first equation constrains the intersection point v to lie on the plane of symmetry. The second and third equation makes the segment $v_1 v_2$ perpendicular to the plane and makes the intersection point v the mid point of segment $v_1 v_2$.

Betweenness Constraint

Betweenness is mostly needed to describe occlusion. To represent betweenness constraints, we need inequations. Inequations can be handled similar to equations by using active sets. A vertex v is between two vertices v_1 and v_2 if and only if there exist two scalars s_1 and s_2 such that:

$$\begin{aligned} s_1 &\geq 0 \\ s_2 &\geq 0 \\ (\mathbf{v} - \mathbf{v}_1) - s_1(\mathbf{v}_2 - \mathbf{v}_1) &= 0 \\ (\mathbf{v}_2 - \mathbf{v}) - s_2(\mathbf{v}_2 - \mathbf{v}_1) &= 0 \end{aligned} \quad (9)$$

2.3 Parse Polyhedra into Nodes and Constraints

The topological and geometric constraints that are implicit in a polyhedral shape are parsed bottom-up into nodes and constraints. This parsing is done locally and as needed.

A vertex v is parsed into a location node \mathbf{v} with three new variables (v_x, v_y, v_z) .

An edge e between two vertices v_1 and v_2 is parsed into a location node e and an orientation node d . The edge location e is the mid point of the two end vertices, and so could be pre-compiled as $0.5(v_1 + v_2)$, and no new variable is needed. The edge orientation d is described by three new variables and three independent nonlinear equations:

$$\begin{aligned} d \times (v_2 - v_1) &= 0 \\ (d \cdot d) - 1 &= 0 \end{aligned} \quad (10)$$

The edge direction is uniquely defined if and only if the two end vertices are distinct. This non singularity condition is enforced by a non zero bound on the distance between the two end vertices.

Let f be a face with boundary edges (e_1, \dots, e_m) and vertices (v_1, \dots, v_n) . Let v_2 be the common vertex between the first two edges e_1 and e_2 . The face location f can be precompiled as $0.5(v_1 + v_3)$ to avoid creating three new variables and generating three linear equations. The face orientation n is described by three new variables and three nonlinear equations:

$$\begin{aligned} n \cdot d_1 &= 0 \\ n \cdot d_2 &= 0 \\ (n \cdot n) - 1 &= 0 \end{aligned} \quad (11)$$

The face normal is uniquely defined if and only if the first two edges are not collinear. This non singularity condition is enforced by bounds on the dot-product of the two edge directions.

If the face f has more than three vertices, the coplanarity of the vertices or edges must be made explicit with additional coplanarity constraints. The smallest number of independent equations is achieved by generating coplanarity constraints from the last $n - 3$ vertices and the face, using equation (6).

2.4 Constrained Minimization

Let x be the vector of variables collected from all the location, orientation, and measure nodes. Let $h(x)$ be the vector of constraint polynomials, collected from all the geometric constraints applied to the model. The shape described by x is feasible if and only if:

$$h(x) = 0 \quad (12)$$

or that the residuals of the constraint equations are all zero.

Geometric models are generally underconstrained, and need to be fit in a least square error sense to data, which could be 2D images or 3D sketches. To fit a 3D model to 2D images, we need camera and projection equations [Slama, 1980]. Let P_k be the camera projection which maps 3D entities in the model represented by x to corresponding 2D entities in the k th image represented by p_k . Let \bar{p}_k be the observed values of p_k given by the k th image. Let W_k be a diagonal matrix of weights $1/\sigma_{k,i}^2$, where $\sigma_{k,i}$ is the standard deviation for the measurement error of the i th variable in the k th image [Press *et al.*, 1988].

The error polynomial is derived from the correspondence between 3D and 2D entities, and the camera projections P_k 's as:

$$\begin{aligned} f(x) &= \sum_k (p_k - \bar{p}_k)^T W_k (p_k - \bar{p}_k) \\ p_k &= P_k(x) \end{aligned} \quad (13)$$

The cameras can be included in the fit by inserting the camera parameters in the model parameters x .

The search for a feasible and best-fit model x is formalized as a constrained minimization problem:

$$\begin{aligned} \text{Minimize } & f(x) \\ \text{subject to } & h(x) = 0 \\ & x_L < x < x_H \end{aligned} \quad (14)$$

The lower and higher bounds (x_L, x_H) are needed to avoid local singularities in location and orientation constraints.

A realistic model can have thousands of nodes. The geometric constraints are local since they usually relate three and at most four nodes each. The ratio of the number of non-zeros versus the number of elements in the Jacobian is much less than 1%. So sparse matrix techniques [Duff, 1981, Bunch and Rose, 1976] must be used to speed up the solution of the constrained shape.

3 Solving Constraint Minimization Problem

The constraint minimization problem (14) can be solved, in theory, using purely symbolic methods or purely numerical optimization techniques. Current symbolic methods are quite complicated to implement and are practically infeasible because of exponential growth in the size of polynomial equations. Numerical optimization techniques are not exact, can be unstable due to numerical errors, or may fail to converge or become trapped at a local minimum. The performance in either case is not very satisfactory. In this section, we develop a hybrid approach which makes use of recent advances in symbolic techniques. Our framework will allow us to exploit advances made in symbolic methods for solving nonlinear constraints and replace numerical steps by symbolic steps as they become feasible.

Our approach has two main steps:

- **Elimination:** Reduce the dimension of the constraint problem by minimizing the number of variables which must be freely varied. We will call them the *independent* parameters, and the remaining parameters as *dependent*. If a dependent parameter is related to other parameters by a linear constraint, then it can be eliminated from the constraints as well as in the error function. If a dependent parameter is related to other parameters by a nonlinear constraint, then root isolation techniques and/or the Newton-Raphson method can be employed to find the feasible values of the dependent parameters satisfying these constraints.
- **Best Fit.** Search for a feasible set of independent parameters which minimizes the error function. It

is also necessary to ensure that any inequality constraints are satisfied. Inequalities arise from user supplied tolerances as well as the need to isolate singularities in the constraint system. The gradient of the error function (its first derivative with respect to each independent parameter) must vanish, which generates additional relationships among the independent parameters in the neighborhood of a minimum of the error function.

3.1 Symbolic Approach

Both of the above steps can be performed by symbolic algorithms from the elimination theory [van der Waerden, 1950] and the theory of real closed fields [Arnon *et al.*, 1984]. Algorithms for solving nonlinear equations such as univariate resultant [Knuth, 1980, Loos, 1982], Macaulay's multivariate resultant [Macaulay, 1916], Buchberger's Gröbner basis algorithm [Buchberger, 1985], and Ritt-Wu's *characteristic set (triangulation)* algorithm [Ritt, 1950, Wu, 1984] can be used for eliminating dependent parameters in the first step. In the process of eliminating dependent parameters, these algorithms can be used to identify any possible inconsistency among constraints. We believe it should be possible to develop heuristics which can be used to localize the nature of inconsistency and identify a subset of constraints causing inconsistency.

The minimization step also can be done symbolically by adding the conditions corresponding to the vanishing of the gradient of the error function with respect to each independent parameter and the positive definiteness of the Hessian of the error function. This system of equations, in conjunction with the constraint equations can determine all of the solutions using root isolation techniques [Collins and Loos, 1982], including the global minimum. Techniques for handling infinitely many solutions describing a surface over which the gradient vanishes may need to be employed.

The symbolic approach is very attractive in that it enumerates all the local minima exactly and finds the global minimum. For this approach to be feasible, it is necessary to develop practical methods for solving two computational problems: (i) computing a triangular form of nonlinear equations (triangular forms are defined later), and (ii) finding solutions of a triangular form by performing arithmetic on algebraic real numbers. Current solutions to both problems require exorbitant computer resources (time and space). We have only limited experience with methods for the second problem, however we have done considerable experimentation with methods for the first problem [Kapur and Mundy, 1989]. We have implementations of Gröbner basis algorithm as well as Ritt-Wu's triangulation algorithm. We have observed that the performance of these algorithms is very sensitive to the degree of overlapping (number of common variables) among equations. Consequently, these algorithms have a very bad performance in computing a triangular form of constraints combined with the relationships arising due to the vanishing of the gradient.

3.2 Numerical Approach

The constraint minimization problem is the same as the general nonlinear programming problem for which numerical techniques have been proposed [Luenberger, 1984]. In the neighborhood of a point in the parameter space, parameters are classified into dependent and independent parameters dynamically by approximating nonlinear constraints by their first-order approximation. A feasible solution on the constraint surface is first computed using the Newton-Raphson method. Ill-conditions are avoided by pivoting about the largest pivots [Strang, 1986, Press *et al.*, 1988].

The error function is expanded in a Taylor series around the feasible solution usually up to second-order. The feasible solution is used to compute a minimum of the error function in its neighborhood by moving in a direction which appears to most rapidly decrease the error function. However, the convergence is only local and can be easily stuck in a weak local minimum. The search is a primal method which works locally in the space of independent parameters using Levenberg-Marquardt technique. Primal methods have faster convergence than penalty and barrier methods, and are simpler to implement than recursive quadratic programming [Luenberger, 1984]. They however require that the search points remain within the feasible region, which is not so difficult since the constraint equations have at most degree two. As the search moves from one feasible region to another, the classification of parameters into independent and dependent parameters also changes. We have recently implemented this numerical approach and the details of the approach are given in Section 4. Our experience suggests that the numerical approach works quite well on medium-sized examples.

3.3 Hybrid Approach

The equations resulting from constraints and the vanishing of the gradient of the error function have considerable overlap, i.e., all variables appear in the error function and consequently, the equation from the vanishing of the gradient are likely to have most of the variables. As a result, algorithms for solving nonlinear constraints, such as the Gröbner basis algorithm and the triangulation algorithm become quite impractical for solving such equations.

Our experience is that the elimination step of reducing the dimension of the constraint problem by eliminating variables can be done effectively using symbolic methods. The second step of finding a best fit between observed values and the model by minimizing the error function can be performed more effectively using numerical optimization techniques.

Our current strategy is to preprocess the constraints and solve them as much as possible using symbolic methods before applying numerical methods. It is beneficial to eliminate as many variables as possible and bring the constraints to a triangular form. In particular, any inconsistency among constraints can be identified in this process. It also appears that symbolic methods can exploit any compositionality property of geometric modeling operators that can be identified. This can lead

to incremental solutions of constraint components and combining these solutions to get a solution for the whole model.

In the remainder of this section, we discuss methods for solving constraints. We give an informal overview of Ritt-Wu's triangulation algorithm with an illustration. For more details, the reader can refer to [Kapur and Mundy, 1989, Wu, 1984, Chou, 1985]. In the final subsection, we discuss a modification of the triangulation procedure which is helpful in controlling the growth of intermediate polynomials in the algorithm.

3.4 Solving Constraints

Linear Constraints

Linear equational constraints can be brought to a triangular form using the Gauss-Jordan method. These constraints can be eliminated by substituting in the remaining constraints and the error function in the dependent parameters in terms of the other parameters.

Nonlinear Constraints

We have experimented with the Gröbner basis algorithm and the triangulation algorithm for solving nonlinear constraints. Both algorithms have worked very well for plane Euclidean geometry theorem proving (see [Chou, 1985, Kapur and Mundy, 1989]). These algorithms have also been used to identifying inconsistency among equations [Kapur and Mundy, 1989, Kapur and Wan, 1990]. In our experience, we have found the triangulation algorithm to exhibit better performance experimentally than the Gröbner basis algorithm using the lexicographic ordering. Consequently, we have been using the triangulation algorithm for solving nonlinear equations.

Given a classification of variables into independent parameters $\{z_1, \dots, z_k\}$ and dependent parameters $\{y_1, \dots, y_l\}$ and a total ordering on dependent parameters, say $y_1 < y_2 < \dots < y_l$, the triangulation algorithm produces from a consistent set of equations, a set of nonlinear equations in the following form:

$$\begin{aligned} g_1(z_1, \dots, z_k, y_1, \dots, y_{i_1}) &= 0 \\ g_2(z_1, \dots, z_k, y_1, \dots, y_{i_2}) &= 0 \\ &\vdots \\ g_m(z_1, \dots, z_k, y_1, \dots, y_{i_m}) &= 0 \end{aligned} \quad (15)$$

where $1 \leq i_1 < i_2 < \dots < i_m \leq l$. The polynomial g_1 is said to be the equation constraining the dependent parameter y_{i_1} ; similarly g_1 and g_2 constraint y_{i_1} and y_{i_2} and so on. Usually, $m = l$ and $i_1 = 1, i_2 = 2, \dots, i_m = l$, however there can be cases where one of the dependent parameters may be skipped implying there is no equation constraining it. Henceforth, we shall assume that $i_1 = 1, i_2 = 2, \dots, i_m = l$, that is, there is an equation constraining every dependent parameter. Polynomial equations in (15) are said to be in *triangular form*. For a discussion about identifying independent and dependent parameters in algebraic equations arising from geometric modeling operations, the reader may consult [Mundy *et al.*, 1989].

For instance, consider the constraints arising from requiring two edges of equal length l on a plane

incident on three distinct vertices with coordinates $(u_0, v_0), (u_1, v_1), (u_2, v_2)$ with orientation vectors B_1, B_2 to be perpendicular to each other.

$$\begin{aligned} h_1 : B_{1,u}^2 + B_{1,v}^2 &= 1 \\ h_2 : B_{2,u}^2 + B_{2,v}^2 &= 1 \\ h_3 : B_{1,u}B_{2,u} + B_{1,v}B_{2,v} &= 0 \\ h_4 : (u_1 - u_0)B_{1,v} + (v_1 - v_0)B_{1,u} &= 0 \\ h_5 : (u_1 - u_0)^2 + (v_1 - v_0)^2 &= l^2 \\ h_6 : (u_2 - u_0)B_{2,v} + (v_2 - v_0)B_{2,u} &= 0 \\ h_7 : (u_2 - u_0)^2 + (v_2 - v_0)^2 &= l^2 \end{aligned} \quad (16)$$

The algorithm described in [Kapur and Mundy, 1989] can be used to triangulate the above constraints. The independent parameters are chosen to be $\{l, u_0, v_0, B_{1,u}\}$. To eliminate a given variable, say y , a polynomial of minimal degree in y is used to pseudo-divide all other polynomials in which y appears. If the pseudo-division does not result in any non-zero remainders in which y appears, then y is considered to be eliminated. Otherwise the pseudo-division operation is repeated on the original polynomials as well as additional polynomials in y obtained as remainders. A triangular form for the above constraints using the ordering $B_{1,v} < B_{2,v} < B_{2,u} < v_1 < u_1 < v_2 < u_2$ on dependent parameters is:

$$\begin{aligned} g_1 : B_{1,v}^2 + B_{1,u}^2 &= 1 \\ g_2 : (B_{1,u}^2 + B_{1,v}^2)B_{2,v}^2 &= B_{1,u}^2 \\ g_3 : B_{1,u}B_{2,u} + B_{1,v}B_{2,v} &= 0 \\ g_4 : (B_{1,u}^2 + B_{1,v}^2)(v_1 - v_0)^2 &= B_{1,v}^2 l^2 \\ g_5 : (u_1 - u_0)B_{1,v} + (v_1 - v_0)B_{1,u} &= 0 \\ g_6 : (B_{2,u}^2 + B_{2,v}^2)(v_2 - v_0)^2 &= B_{2,v}^2 l^2 \\ g_7 : (u_2 - u_0)B_{2,v} + (v_2 - v_0)B_{2,u} &= 0 \end{aligned} \quad (17)$$

The algorithm discussed in [Kapur and Mundy, 1989] eliminates dependent parameters in the descending order starting with polynomials in the highest dependent parameter. One can eliminate variables in the ascending order starting with the lowest dependent parameter which is likely to produce a simpler triangular form. However, this change in the algorithm worsens its performance. Instead it might be better to obtain a triangular form by eliminating dependent parameters in the descending order first and then process the *initials* (the leading coefficients) of the polynomials in the result in the ascending order. The simplified triangular form thus obtained from the above triangular form is:²

²The above discussion of computing a triangular form is quite informal. There are many technical details which have been omitted. In particular, we have not discussed degenerate conditions generated by the triangulation algorithm; we have also not discussed the case when a triangular form is not irreducible. For a comprehensive discussion and technical details, an interested reader can consult [Wu, 1984, Chou, 1985].

$$\begin{aligned}
g_1: & B_{1,v}^2 + B_{1,u}^2 &= 1 \\
g_2': & B_{2,v}^2 - B_{1,u}^2 &= 0 \\
g_3: & B_{1,u}B_{2,u} + B_{1,v}B_{2,v} &= 0 \\
g_4': & (v_1 - v_0)^2 - B_{1,v}^2 l^2 &= 0 \\
g_5: & (u_1 - u_0)B_{1,v} + (v_1 - v_0)B_{1,u} &= 0 \\
g_6': & B_{1,u}^2(v_2 - v_0)^2 &= B_{1,u}^2 B_{2,v}^2 l^2 \\
g_7: & (u_2 - u_0)B_{2,v} + (v_2 - v_0)B_{2,u} &= 0
\end{aligned} \tag{18}$$

Polynomial equation g_6' can be further simplified by removing the common factor $B_{1,u}^2$ by requiring that $B_{1,u}$ be nonzero. Other heuristics discussed in [Kapur and Wan, 1990] as well as recently by Chou can be used to speed-up the triangulation algorithm.

The reader would have noted that a triangular form (15) obtained from nonlinear constraints is similar to a triangular form obtained from linear constraints using the Gaussian elimination algorithm, with the main difference being that in the above case, the constraint defining a variable is nonlinear. (In fact, if given linear constraints, the triangulation algorithm works the same way as the Gauss-Jordan algorithm.) Consequently, it is not possible to just do back-substitution for the dependent parameters. Instead, in (15), there is a need to solve $g_1 = 0$ for y_1 in terms of z_1, \dots, z_k . For each solution s for y_1 , substitute s for y_1 in g_2 . Solve $g_2[y_1 \leftarrow s]$ for y_2 , and so on. Methods for doing arithmetic on algebraic real numbers can be used [Collins and Loos, 1982]. The degrees of the polynomials in triangular form often grow quite fast because of which root finding could be very ill-conditioned.

During the triangulation algorithm, it is not uncommon to encounter large number of large polynomials. Because of this blow-up in the growth of intermediate polynomials, computation of a triangular form can require considerable computational resources (space and time). In the next subsection, we propose some heuristics for dealing with the problem of large intermediate polynomials. These heuristics have been implemented and they appear to be quite promising.

3.5 Handling Big Coefficients in Triangulation

In applying the triangulation algorithm for processing geometric constraints for modeling and eliminating variables, we have often observed that the pseudo-division of a polynomial with another polynomial can be a very expensive operation. As a result, even though a triangular form of a set of constraints may not be that big, but the intermediate polynomials become very big. This is so because polynomials are represented in recursive form and the coefficients (which themselves are polynomials) can be quite big. In order to circumvent the problem, we have modified the triangulation algorithm given on p. 21 in [Kapur and Mundy, 1989]. We will not give the new triangulation algorithm here, but instead discuss what steps in the algorithm must be modified and illustrate these changes with examples. The modified algorithm with full technical details will be discussed in a forthcoming paper.

The main idea is to replace big coefficients (polynomials with more than two terms say) of terms made up of

dependent parameters in polynomials by new variables. The bindings (substitutions) of these new variables are stored for subsequent computations. This replacement is performed on the original polynomials as well as any intermediate polynomials generated. Pseudo-reduction computations are performed on these new polynomials in which the coefficients of dependent terms are simple.

In step 2 of the triangulation algorithm on p. 21 in [Kapur and Mundy, 1989], when a polynomial with the minimal degree of the parameter being eliminated is selected, it is checked whether its leading coefficient (initial) is non-zero. If the initial is zero, then the polynomial is simplified by deleting its leading monomial. The check is repeated again on the leading coefficient of the simplified polynomial. When it is ascertained that the leading coefficient is indeed non-zero, then the polynomial is marked to remember that it has a non-zero leading coefficient and it has been used for pseudo-dividing other polynomials in T . If the polynomial becomes identically zero after simplification, then that polynomial is removed from T and step 2 is repeated. If the polynomial thus selected is already marked, this implies that it has already been used for pseudo-division and the pseudo-division of other polynomials in T by this polynomial gives remainder which are in lower variables; so, that polynomial is inserted into G , the triangular form.

There are many ways to check whether the initial of a polynomial is non-zero or not. Of course, bindings for the new variables in the initial can be recursively substituted giving an expression in original parameters. This expression can be simplified and checked to be non-zero. However, simplification is likely to be quite expensive, especially if the initial is non-zero. Instead, for each of the independent parameters, some random rational values are selected and the initial is computed for these values of the independent parameters (for speeding the computation of the value of the initial, p-adic arithmetic could be used instead of arbitrary precision rationals). If the value of the initial is found to be zero, some other values for the independent parameters are randomly selected. If the initial keeps evaluating to 0 on a certain number of tries (say 4), then the initial can be simplified by recursively substituting for the new variables. If the value of the initial is non-zero for any set of values of independent parameters, then the initial is not identically equal to zero.

We illustrate these modification using an example. Consider a torus illustrated in figure 2. The constraints defining the torus are:

$$\begin{aligned}
1.: & rs \cos u \cos t + rl \cos t &= x \\
2.: & rs \cos u \sin t + rl \sin t &= y \\
3.: & rs \sin u &= z \\
4.: & \cos u^2 + \sin u^2 &= 1 \\
5.: & \cos t^2 + \sin t^2 &= 1
\end{aligned} \tag{19}$$

The objective is to compute an implicit representation of the torus in terms of x, y, z and the parameters rs , the smaller radius, and rl , the outer radius. This can be achieved by successively eliminating $\cos t$, $\sin t$, $\sin u$ and $\cos u$ from the above equations.

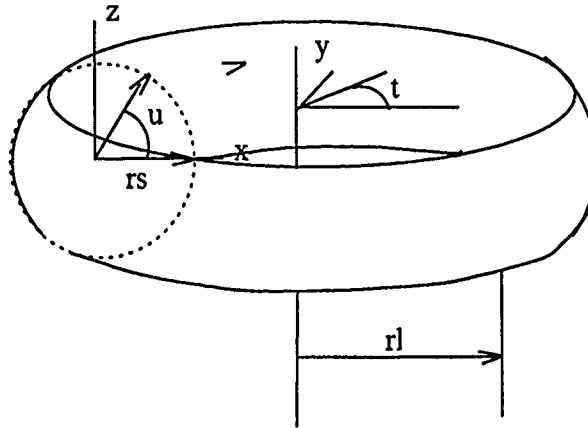


Figure 2: Torus.

The variable *cost* can be eliminated by pseudo-dividing the polynomial from constraint 5 by the polynomial from constraint 1, which gives as remainder

$$6. \quad rs^2 \cos u^2 \sin t^2 + 2rl \, rs \cos u \sin t^2 + rl^2 \sin t^2 - rs^2 \cos u^2 - 2rl \, rs \cos u - rl^2 + x^2.$$

The variable *sint* is eliminated by pseudo-dividing polynomial 6 by the polynomial from constraint 2. The remainder is:

$$7. \quad -rs^4 \cos u^4 - 4rl \, rs^3 \cos u^3 + (rs^2 x^2 + rs^2 y^2 - 6rl^2 rs^2) \cos u^2 + (2rl \, rs x^2 + 2rl \, rs y^2 - 4rl^3 rs) \cos u + (rl^2 x^2 + rl^2 y^2 - rl^4).$$

Similarly, *sinu* can be easily eliminated from the polynomial in constraint 4 by the polynomial in constraint 3; the remainder is a polynomial with 3 terms. Then *cosu* can be eliminated using the result of the last step from polynomial 7 above. The result of these eliminations are two polynomials in *x, y, z, rs, rl* with one polynomial having over 300 monomials. In order to get the result, this polynomial has to be pseudo-divided using the other polynomial which takes approximately 10 minutes on a Symbolics 3600 Lisp machine.

Using the modified algorithm, after polynomial 7 above is generated, it is replaced by the polynomial 7' by replacing big coefficients by new variables:

$$7'. \quad -rs^4 \cos u^4 - 4rl \, rs^3 \cos u^3 + A_1 \cos u^2 + A_2 \cos u + A_3$$

where

$$A_1 = rs^2 x^2 + rs^2 y^2 - 6rl^2 rs^2$$

$$A_2 = 2rl \, rs x^2 + 2rl \, rs y^2 - 4rl^3 rs$$

$$A_3 = rl^2 x^2 + rl^2 y^2 - rl^4.$$

As a result, after eliminating *sinu* and *cosu* from the polynomials, only two polynomials with 3 and 6 terms are generated. And, the polynomial with 3 terms is the desired answer which can be found in a few seconds.

The result given by the triangulation algorithm for the torus has an extraneous factor, similar to the case of g'_6 in the previous subsection. The implicit representation

$$4 \, rl^2 (rs^2 - z^2) = (x^2 + y^2 + z^2 - rl^2 - rs^2)^2$$

is obtained by removing this common factor.

The gains of the above modification are quite significant for larger examples. There are examples which could not be completed even after 10 hours of computing time, whereas they could be done in a few minutes using the modified algorithm. More examples and technical details of the modified triangulation algorithm will be discussed in a forthcoming paper.

4 The Numerical Constraint Solution Algorithm

The following section discusses a primarily numerical algorithm for the solution of constraint equations. Symbolic methods enter in the recompilation of constraints into algebraic expressions.

4.1 Algorithm and Example

1. Consistent linear equations in (12) are eliminated through Gaussian elimination and variable substitution. Example is the pre-compilation of edge or face locations as linear combinations of vertex locations.

Nonlinear equations in (12) are solved by Newton-Raphson iterations with the Jacobian derived symbolically from the constraint equations. Since the polynomial equations have at most second order, parabolic interpolation makes the line search for minimum residual very fast and accurate. The Jacobian is non singular except at local singularities due to being stuck at zero length or angle. These local singularities are avoided by putting bounds on the length and angle measures.

2. Fitting or minimizing the sum of square errors is done in the space of free variables. The decomposition into free and dependent variables is revealed through finding the largest non singular submatrix of the Jacobian. The gradient and Hessian polynomials are computed symbolically from the error polynomial (13), projected onto the plane tangent to the constraint surfaces, and expressed in the reduced space of free variables.

Levenberg-Marquardt iterations vary continuously between steepest descent and inverse Hessian to find values for the free variables such that the fitting error is minimized. Goodness of fit and error estimates for the free variables can be deduced from the reduced Hessian at the local minimum found.

Figures 3 and 4 illustrate the convergence of a twisted and broken-up airplane to a feasible shape satisfying polyhedral, attachment, and symmetry constraints, and then to a final shape least-square fit to views of a specific airplane model. The constrained problem has 263 variables, 276 equations, and about 32 degrees of freedom. The first feasible solution is found after 4 iterations, and the best-fit solution after 5 more iterations. Figure 3 shows the convergence to a feasible shape (iterations: 0, 1, 2) and figure 4 shows the convergence to a best-fit shape (iterations: 4, 5, 6).

4.2 Newton-Raphson Solves for Feasible Shape

Consider the Taylor expansion of a single constraint polynomial $h(\mathbf{x})$:

$$h(\mathbf{x} + \alpha \mathbf{dx}) = h(\mathbf{x}) + \alpha \nabla h(\mathbf{x}) \mathbf{dx} + \frac{1}{2} \alpha^2 \mathbf{dx}^T \nabla^2 h(\mathbf{x}) \mathbf{dx} + \dots \quad (20)$$

Unless $h(\mathbf{x})$ is singular, i.e. $\nabla h(\mathbf{x}) = \mathbf{0}^T$, α can always be chosen small enough so that the first order term is dominant relative to the second and higher order terms, and the residual of $h(\mathbf{x})$ is driven to zero.

A line search along the direction \mathbf{dx} finds α such that the residual $h(\mathbf{x} + \mathbf{dx})$ is a local minimum. The line search is implemented by a bisection search followed with a parabolic interpolation. The bisection search shrinks the increments \mathbf{dx} in half each time, and finds three points $(\mathbf{x}, \mathbf{x} + 0.5\mathbf{dx}, \mathbf{x} + \mathbf{dx})$, such that the residual at the mid point is strictly less than the residuals at the two extreme points. Then a parabolic interpolation through the three points finds the local minimum. Parabolic interpolation gives very accurate local minimum because the constraint polynomial $h(\mathbf{x})$ has maximum order two.

The direction of increment \mathbf{dx} is normal to all the constraint surfaces, and is computed from the first order terms of the constraint equations (12):

$$\nabla h(\mathbf{x}) \mathbf{dx} = -h(\mathbf{x}) \quad (21)$$

System (21) is only a linear approximation of the nonlinear equations in the neighborhood of point \mathbf{x} . This approximation makes linear system (21) inconsistent when there are redundant constraint equations and point \mathbf{x} is far from the feasible surfaces [Nelson, 1985]. However, the least square error is small. Least-norm solution must

also be solved because there are usually fewer independent equations than unknowns. Least-norm and least-square-error are consistent with finding the nearest most feasible shape. The least-norm and least-square-error solution \mathbf{dx} can be computed from a singular value decomposition of the Jacobian matrix $\nabla h(\mathbf{x})$ [Strang, 1986, Press *et al.*, 1988].

Convergence to the nearest feasible shape is done using Newton-Raphson iterations. Far from the constraint surfaces, the speed of convergence is dictated by how successful the line search is. The bisection search has linear convergence ratio 0.5 with the fewest number of polynomial evaluations. The bisection search is further made adaptive by continuously varying the shrink factor. As the surfaces become close, the residuals and increments become small and line search is not needed. Newton-Raphson iterations have quadratic convergence.

Iterative numerical methods can only assure local convergence. Even local convergence is hard to achieve because we have to watch out for singularities in the Jacobian. The Jacobian have rows the gradients of the constraint equations. The Jacobian becomes singular when the number of independent gradients decrease because some gradients become either zero or dependent. The singular location constraint has zero gradient, equation (2). It is improperly represented by a second order equation instead of three first order equations, and the root has multiplicity two instead of one. The singular orientation constraint has gradient dependent to the gradients of the normalization constraints, equations (4). Both singular value decomposition and least-norm solution throw away singularities. The net effect is similar to dropping out the singular constraints, and so their residuals are left unchanged.

For most singular cases, the Jacobian is ill-conditioned with a very large condition number. To relate the condition number of the Jacobian matrix with the presence of singularities, the increments in \mathbf{dx} must have comparable scales. The orientation vectors have unit magnitudes and so their coordinates varies at most by 1. Similarly, $\cos \alpha$ is in the open interval $]-1, +1[$ and varies at most by 1. The size of the model must be normalized so that the location and length increments varies at most by 1. With a normalized model, the residuals and the partial derivatives of the constraint equations have scales comparable to 1. Local singularities in location and orientation constraints are easily avoided by choosing a pivot threshold much smaller than the threshold for nonzero length and angle.

An equation is redundant (resp. inconsistent) if and only if it can be expressed symbolically as a linear combination of independent equations and a zero (resp. non zero) constant. This is equivalent to a redundant (resp. inconsistent) linear system (21) in the increment unknowns \mathbf{dx} if and only if the equations have order less than two. For equations with order two or higher, linear combinations of symbolic expressions implies linear combinations of gradients evaluated at any point \mathbf{x} , but not vice versa [Light and Gossard, 1982, Serrano and Gossard, 1987].

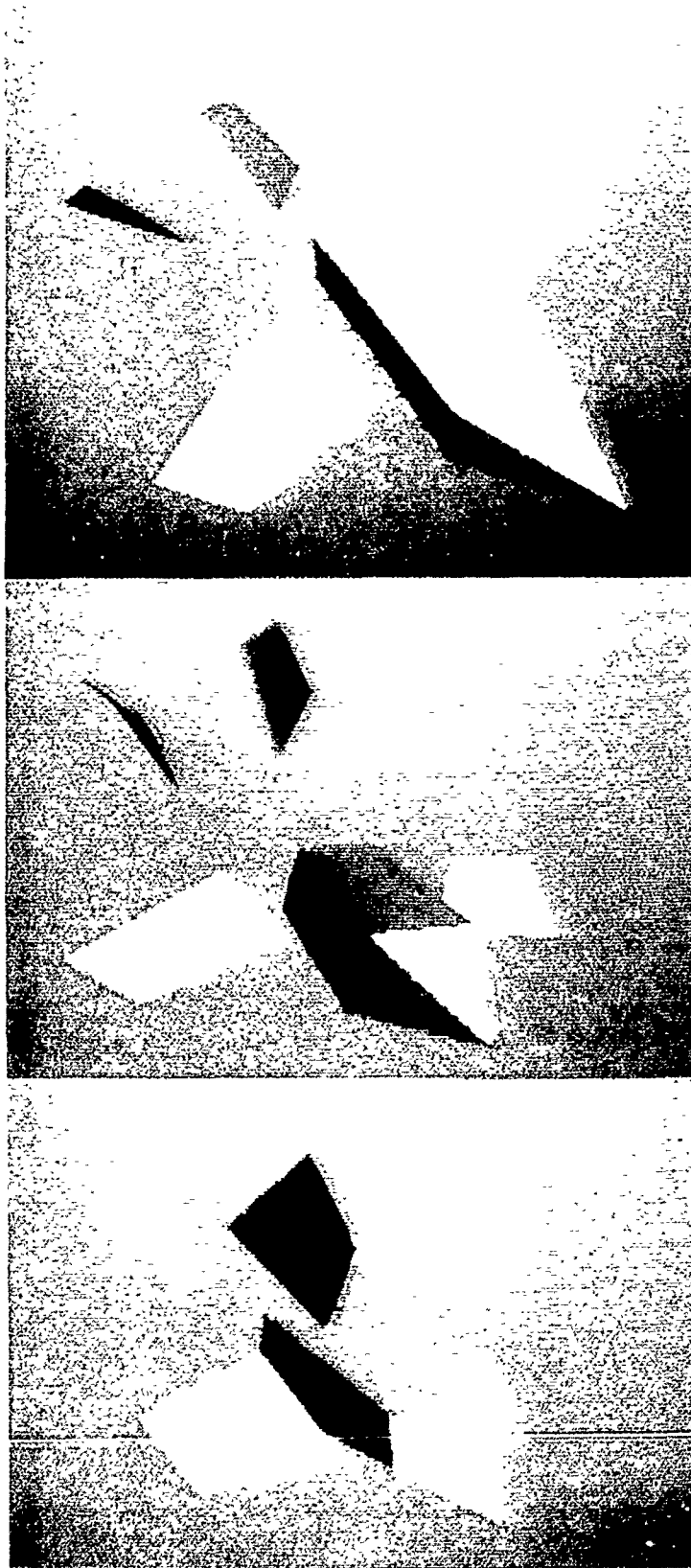


Figure 3: Search for feasible shape.

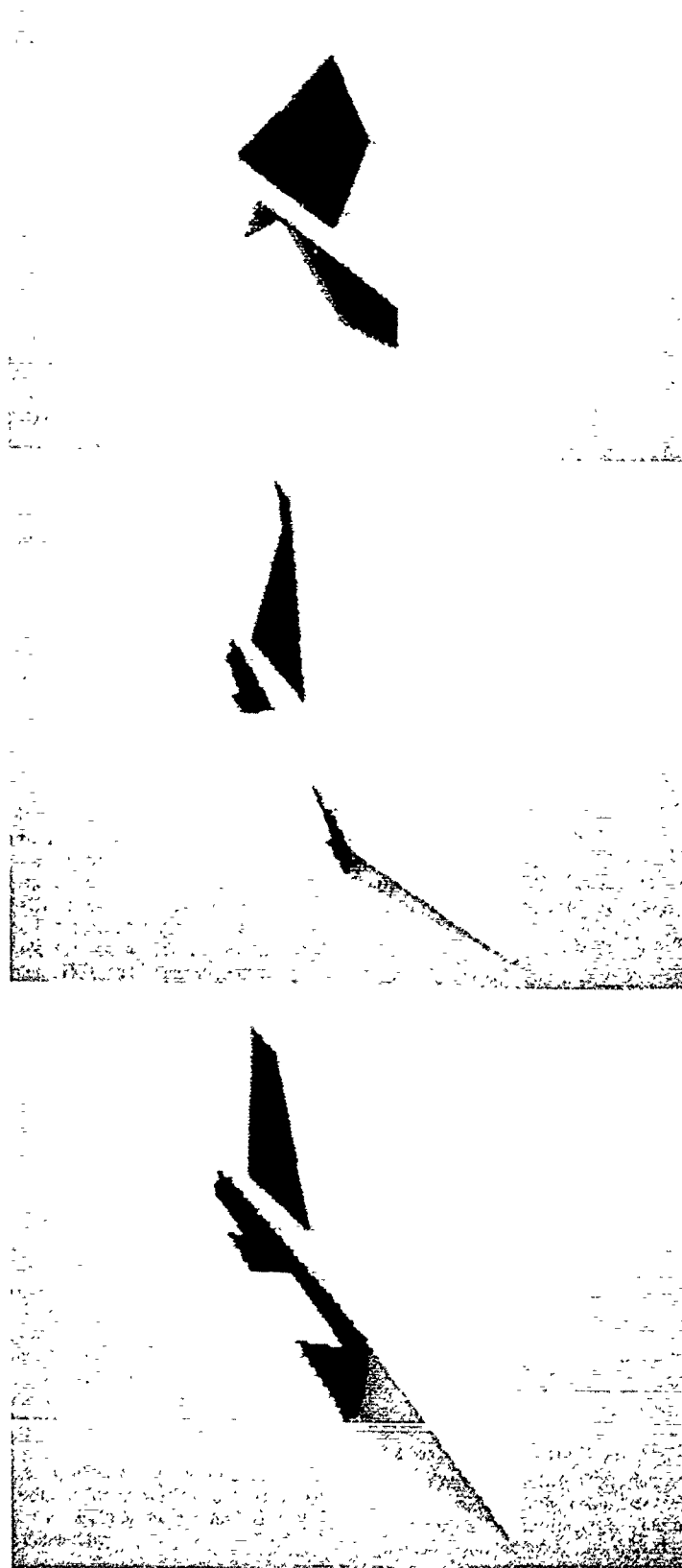


Figure 4: Search for best-fit shape.

4.3 Reduced Space of Free Variables

Let the Jacobian $\nabla h(\mathbf{x})$ of the constraint equations be split into two submatrices $\nabla_y h(\mathbf{y}, \mathbf{z})$ and $\nabla_z h(\mathbf{y}, \mathbf{z})$, such that the submatrix $\nabla_y h(\mathbf{y}, \mathbf{z})$ is non singular. The variables \mathbf{x} are correspondingly split into dependent variables \mathbf{y} , and free variables \mathbf{z} . Increments tangent to the constraint surfaces are described by linear system (21), where the residuals $h(\mathbf{x})$ are zeros, or by:

$$\begin{aligned} d\mathbf{y} &= [dy/dz] dz \\ [dy/dz] &= - [\nabla_y h(\mathbf{y}, \mathbf{z})]^{-1} [\nabla_z h(\mathbf{y}, \mathbf{z})] \end{aligned} \quad (22)$$

The decomposition between dependent and free variables is done during convergence, instead of being fixed ahead of time from the dependency of the nodes [Mundy *et al.*, 1989, McClain, Proposal]. Gaussian elimination, with row and column pivotings about largest pivots first, transforms the numerical Jacobian matrix $\nabla h(\mathbf{x})$ into an echelon form with maximum rank $n - r$. The dependent variables correspond to the first $n - r$ columns, whose pivots are non zero. The free variables correspond to the last r columns whose pivots are almost zero.

Using partial derivatives respective to the variables in \mathbf{y} and \mathbf{z} , and substituting the dependent increments with equation (22), the first order term in the Taylor expansion of the error function f is:

$$\begin{aligned} df &= \nabla_y f d\mathbf{y} + \nabla_z f d\mathbf{z} \\ &= (\nabla_y f [dy/dz] + \nabla_z f) d\mathbf{z} \end{aligned} \quad (23)$$

The reduced gradient, denoted by $D_r f$, is the projection of the full gradient ∇f onto the subspace of increments tangent to the constraint surfaces, expressed in the space of the free variables \mathbf{z} :

$$(D_r f) = \nabla_y f [dy/dz] + \nabla_z f \quad (24)$$

Similarly, the second order term in the Taylor expansion of the error function f is:

$$\begin{aligned} d^2 f &= \frac{1}{2} (d\mathbf{y}^T d\mathbf{z}^T) \begin{pmatrix} \nabla_{yy}^2 f & \nabla_{yz}^2 f \\ \nabla_{zy}^2 f & \nabla_{zz}^2 f \end{pmatrix} \begin{pmatrix} d\mathbf{y} \\ d\mathbf{z} \end{pmatrix} \\ &= \frac{1}{2} d\mathbf{z}^T \left([dy/dz]^T \nabla_{yy}^2 f [dy/dz] + \nabla_{zz}^2 f \right) \\ &\quad + d\mathbf{z}^T \nabla_{zy}^2 f [dy/dz] d\mathbf{z} \end{aligned} \quad (25)$$

The reduced Hessian, denoted by $D_r^2 f$, describes the restriction of the full Hessian $\nabla^2 f$ to the subspace of increments tangent to the constraint surfaces, expressed in the space of free variables \mathbf{z} :

$$\begin{aligned} D_r^2 f &= [dy/dz]^T \nabla_{yy}^2 f [dy/dz] + \nabla_{zz}^2 f \\ &\quad + 2 \nabla_{zy}^2 f [dy/dz] \\ &\approx [dy/dz]^T \nabla_{yy}^2 f [dy/dz] + \nabla_{zz}^2 f \end{aligned} \quad (26)$$

The eigenvalues of the reduced Hessian must be strictly positive to insure convergence to a local minimum.

The minimization function $f(\mathbf{x})$ is the sum of squares of the errors between the actual and observed shapes, and so is globally convex. The convexity of $f(\mathbf{x})$ makes

the partial Hessians $\nabla_{yy}^2 f$ and $\nabla_{zz}^2 f$ positive semi-definite. The positive semi-definite Hessian $\nabla_{yy}^2 f$ transformed to the space of free variables \mathbf{z} by pre-multiplying by the transpose of $[dy/dz]$ and post-multiplying by $[dy/dz]$ is always positive semi-definite [Strang, 1986]. During the convergence, the matrix $\nabla_{zy}^2 f [dy/dz]$ can be non positive definite and ruin the positive definiteness of the reduced Hessian, and so is dropped out of the approximation. Note that the matrix $\nabla_{zy}^2 f$ contains off-diagonal terms of the full Hessian and is mostly zeros anyway.

The increments $d\mathbf{z}$ are solved so as to minimize the fit between actual and observed data, expressed in the reduced space of free variables \mathbf{z} :

$$(D_r^2 f) d\mathbf{z} = -(D_r f)^T \quad (27)$$

Then the increments $d\mathbf{y}$ are deduced from equation (22) so as to stay on the plane tangent to the constraint surfaces. However, since the constraint equations are non-linear the tangent plane is only a local first order approximation to the constraint surfaces. To insure feasibility, we need to return to the constraint surfaces by following the old normals, described by equation (22), using a few additional Newton-Raphson iterations.

4.4 Levenberg-Marquardt Solves for Best-Fit Shape

The inverse Hessian method, equation (27), is appropriate only when close to the minimum. Far from the minimum, steepest descent method must be used. Levenberg-Marquardt iterations use a scale λ to switch continuously from steepest descent to inverse Hessian, as the minimum is approached [Press *et al.*, 1988]:

$$(D_r^2 f + \lambda \text{Diag}(D_r^2 f)) d\mathbf{z} = -(D_r f)^T \quad (28)$$

Large λ makes the Hessian diagonally dominant, and the iteration becomes steepest descent with a line search along the minus gradient:

$$\lambda \text{Diag}(D_r^2 f) d\mathbf{z} = -(D_r f)^T \quad (29)$$

$\text{Diag}(D_r^2 f)$ contains the diagonal elements of the Hessian. The diagonal elements are approximatively equal to the eigenvalues of the Hessian, and so describe the relative scales of the variables. The zig-zags in ordinary steepest descent are avoided due to the availability of these approximate scales. The revised steepest descent converges with much smaller linear rate $[(A - a)/(A + a)]^2$, where the maximum and minimum eigenvalues, A and a , are almost equal because of the rescaling of the variables with $\text{Diag}(D_r^2 f)$ [Luenberger, 1984].

As the minimum is close, λ goes to zero and the iteration becomes inverse Hessian with quadratic convergence, equation (27). The convergence of the constrained minimization near the local minimum is also quadratic. The constraint surfaces have degree at most two, and so are well approximated by their tangent planes and the least singular mapping $[dy/dz]$.

At the local minimum, the reduced gradient is zero, i.e. the full gradient is perpendicular to the constraint

surfaces. The reduced Hessian, or the full hessian restricted to the tangent space must be positive definite for a strict local minimum. Zero eigenvalues of the reduced Hessian correspond to free variables that can not be deduced by the least-square error fit. Examples could be coordinates of the vertices that are not visible in any images, and are not constrained by any other constraints such as symmetry. At other times, the reduced Hessian is ill-conditioned with nearly zero eigenvalues. This corresponds to a fit to unreliable data with large standard error σ_i .

If the measurement errors are available then the inverse of the reduced Hessian is equal to the estimated covariance matrix of the standard errors in the fitted parameters z . A statistical measure of goodness-of-fit can also be computed [Press *et al.*, 1988].

References

- [Arnon *et al.*, 1984] D.S. Arnon, G.E. Collins, and S. McCallum. Cylindrical algebraic decomposition. i: the basic algorithm, ii: an adjacency algorithm for the plane. *SIAM J. of Computing*, 13:865-877, 878-889, 1984.
- [Brooks, 1981] Rodney Brooks. Symbolic reasoning among 3d models and 2d images. In J.M. Brady, editor, *Computer Vision*. North Holland, 1981.
- [Buchberger, 1985] Bruno Buchberger. Groebner bases: An algorithmic method in polynomial ideal theory. In N.K. Bose, editor, *Multidimensional System Theory*, pages 184-232. D. Reidel Publishing Co., 1985.
- [Bunch and Rose, 1976] James R. Bunch and Donald J. Rose, editors. *Sparse Matrix Computations*. Academic Press, 1976.
- [Chou, 1985] S.C. Chou. *Proving and discovering theorems in elementary geometry using Wu's method*. PhD thesis, University of Texas at Austin, Dept. of Mathematics, 1985.
- [Collins and Loos, 1982] George E. Collins and Ruediger G.K. Loos. Real zeros of polynomials. In Bruno Buchberger, George E. Collins, and Ruediger G.K. Loos, editors, *Computing Supplementum 4: Computer Algebra-Symbolic and Algebraic Computation*. Springer-Verlag, 1982.
- [Duff, 1981] Iain S. Duff, editor. *Sparse Matrices and their Uses*. Academic Press, 1981.
- [Faux and Pratt, 1979] I. D. Faux and M. J. Pratt. *Computational Geometry for Design and Manufacture*. Ellis Horwood Ltd., 1979.
- [Forsyth *et al.*, In preparation] David Forsyth, Joseph L. Mundy, Andrew Zisserman, and Christopher M. Brown. Projectively invariant representations using implicit algebraic curves. In preparation.
- [Horn, 1986] Berthold K. P. Horn. *Robot Vision*. MIT Press, 1986.
- [Kapur and Mundy, 1989] Deepak Kapur and Joseph L. Mundy, editors. *Geometric Reasoning*. MIT Press, 1989.
- [Kapur and Wan, 1990] Deepak Kapur and Hoi K. Wan. Refutational proofs of geometry theorems via characteristic set computation. In *ACM-SIGSAM 1990 International Symposium on Symbolic and Algebraic Computation - ISSAC '90*, Tokyo, Japan, 1990.
- [Knuth, 1980] D.E. Knuth. *Seminumerical algorithms: The art of computer programming*, volume 2. Addison Wesley, second edition, 1980.
- [Light and Gossard, 1982] Robert Light and David Gossard. Modification of geometric models through variational geometry. *Computer Aided Design*, 14(4):209-214, 1982.
- [Loos, 1982] Ruediger G.K. Loos. Generalized polynomial remainder sequences. In Bruno Buchberger, George E. Collins, and Ruediger G.K. Loos, editors, *Computing Supplementum 4: Computer Algebra-Symbolic and Algebraic Computation*. Springer-Verlag, 1982.
- [Luenberger, 1984] David D. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley Pub., second edition, 1984.
- [Macaulay, 1916] F.S. Macaulay. *The Algebraic Theory of Modular Systems*. Cambridge University Press, 1916.
- [McClain, Proposal] Richard A. McClain. *Reconstructing Scene Geometry From Images*. PhD thesis, University of Pennsylvania, Systems Engineering Dept., Proposal.
- [Mundy *et al.*, 1989] Joseph L. Mundy, Patricia Vrobel, and Reuben Joynson. Constraint-based modeling. In *Proc. DARPA Image Understanding Workshop*, pages 425-442, Palo Alto, CA, 1989.
- [Nelson, 1985] Greg Nelson. Juno, a constraint-based graphics system. *ACM Computer Graphics, SIGGRAPH '85*, 19(3):235-243, 1985.
- [Paul, 1981] Richard P. Paul. *Robot Manipulators: Mathematics, Programming, and Control*. MIT Press, 1981.
- [Ponce and Kriegman, 1989] Jean Ponce and David J. Kriegman. On recognizing and positioning curved 3d objects from image contours. In *Proc. DARPA Image Understanding Workshop*, pages 179-188, Palo Alto, CA, 1989.
- [Press *et al.*, 1988] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1988.
- [Ritt, 1950] J.F. Ritt. *Differential Algebra*. AMS Colloquium Publications, 1950.
- [Roberts, 1965] L.G. Roberts. Machine perception of three-dimensional solids. In J.T. Tippett *et al.*, editor, *Optical and Electro-Optical Information Processing*, pages 159-197. MIT Press, 1965.
- [Serrano and Gossard, 1987] D. Serrano and D. Gossard. Constraint management in conceptual design. In D. Sriram and R. A. Adey, editors, *Knowledge Based*

- Expert Systems in Engineering: Planning and Design*, pages 211-224. Computational Mechanics Publications, 1987.
- [Slama, 1980] Chester C. Slama, editor. *Manual of Photogrammetry*. American Society of Photogrammetry, fourth edition, 1980.
- [Strang, 1986] Gilbert Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, 1986.
- [Sugihara, 1986] Kokichi Sugihara. *Machine Interpretation of Line Drawings*. MIT Press, 1986.
- [Sutherland, 1963] I.E. Sutherland. Sketchpad: A man-machine graphical communications system. Technical Report 296, MIT Lincoln Laboratories, 1963. Also published by Garland Publishing Inc, New York, 1980.
- [van der Waerden, 1950] B.L. van der Waerden. *Modern Algebra: Volumes I and II*. Fredrick Ungar Publishing Co, 1950.
- [Wu, 1984] W. Wu. Basic principles of mechanical theorem proving in geometries. *J. of System Sciences and Mathematical Sciences*, 4(3):207-235, 1984. Also published in *J. of Automated Reasoning* 2 (1986).

MARVEL: Location Recognition Using Stereo Vision*

David J. Braunegg[†]
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

To use a world model, a mobile robot must be able to determine its own position in the world. To support truly autonomous navigation, I present MARVEL, a system that builds and maintains its own models of world locations and uses these models to recognize its world position from stereo vision input. MARVEL is designed to be robust with respect to input errors and to respond to a gradually changing world by updating its world location models. I present results from real-world tests of the system that demonstrate its reliability. MARVEL fits into a world modeling system under development.

1 Introduction

To achieve navigation over a long lifetime, a mobile robot needs a memory of the world, i.e., a map or "world model." For true autonomy, the robot must be able to navigate in places of which it has no previous knowledge. Thus, we want the robot to build its world model instead of having it supplied *a priori*. Unfortunately, due to the problem of cumulative error, exact metrical models of the world cannot be used [Brooks, 1985]. The most promising alternative is a topological map that contains world locations and information about how they are connected [Kuipers, 1977] [Chatila and Laumond, 1985]. However, due to errors and uncertainty in odometry, we cannot follow such a map exactly. Thus, we need the ability to recognize the locations contained in such a map.

The first part of the problem we are considering, then, is how to *build* models of world locations and *use* them

*This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the Laboratory's Artificial Intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-85-K-0124 and under Army contract number DACA76-85-C-0010 and in part by the Office of Naval Research University Research Initiative Program under Office of Naval Research contract N00014-86-K-0685.

[†]The author is now with the Image Processing Research Laboratory, The MITRE Corporation, Bedford, MA 01730.

for recognition. The problem becomes more difficult when we realize that the world changes over time. Also, our sensory input is imperfect. Therefore, the second part of the problem is to *maintain* these models over time as the world changes and as we receive new sensory data which is noisy. The implemented system MARVEL (Model building And Recognition using Vision to Explore Locations) addresses these problems by *building*, *using*, and *maintaining* models of world locations for recognition to support navigation.

To recognize a location, we first take a series of stereo pair images from a single position in the current room. The stereo vision module then finds salient features of the room and abstracts them into the representation which will be used for recognition. Recognition is performed by comparing this representation to room models which were built by the system from similar stereo data obtained previously. The results of this recognition are used to update the existing model to reflect the current state of the room and the importance of the features to the recognition. This recognition system fits into a larger modeling system under development. A summary of the results of this research are given here, with a fuller account of the research available in [Braunegg, 1990a].

2 World Features from Stereo Vision

Stereo vision provides the locations of world features in camera-centered world coordinates. Since these locations have relatively good spatial resolution (compared with sonar data, for example), the stereo data is a good candidate for input to the recognition system. (The error bounds on the stereo data have been investigated by [Matthies and Shafer, 1986].)

We use the Marr-Poggio-Grimson stereo algorithm [Marr and Poggio, 1979] [Grimson, 1981] [Grimson, 1985] [Braunegg, 1990b] to obtain our stereo data. This stereo algorithm is based on intensity-edge features in the images. Since such features usually correspond to physical edges of visible objects, they characterize the distribution of objects in the visible world. Employing the heuristic that large objects tend not to move and thus help to identify the specific areas in which they reside (e.g., doorways, windows, bookcases), we eliminate short stereo features from the stereo data. Also, since our camera geometry uses horizontal epipolar lines, the localization of the stereo features deteriorates as the features



Figure 1. Groundplane projection of the vertical stereo features from the full set of stereo pairs for Room 914. The circle shows the camera location and the tick mark in the circle indicates 0° in the camera-based world coordinate system.

become more horizontal.

To get a full view of the room containing the robot, we rotate the robot through 360°, taking overlapping views of the room [Brooks, 1986]. The stereo data from these overlapping views is then "pasted" together for a full representation of the room. Using the camera geometry, the stereo feature coordinates are converted into camera-centered world coordinates and the ceiling and floor features removed.

3 Model/Data Representation

Although it might be tempting to try to create a representation of a room that looks like an architect's floor-plan, this is not possible. For a recognition system, the world is not defined as we would like it to be, but instead is defined by what is observable by the sensors that are being used. Thus, in our case, the representation must be related to the matched stereo features obtained from the stereo algorithm.

Other researchers have investigated the use of full 3-D stereo features for map building [Faugeras *et al.*, 1986] [Braunegg, 1989]. However, our task is different in that we wish to recognize rooms rather than navigate through them. To build our room representation for recognition, we project the vertical room features from the stereo algorithm to the groundplane (Figure 1). We have found that this 2-D representation sufficiently characterizes the room for the purpose of recognition.

The 2-D groundplane representation of the stereo features has the added benefit of reducing the amount of data that must be handled for each location to be recognized. We further abstract the data by using it to develop an occupancy-grid representation of the current room [Moravec and Elfes, 1985]. We impose a grid with 1-foot spacing on the floor of the room and mark each grid square that has a vertical stereo feature falling in it (Figure 2).

The final representation of a room, then, is a set of grid squares that mark the locations of vertical room features as determined by the stereo algorithm. The locations of these squares are described in terms of a camera-centered coordinate system. The orientation of this coordinate system is determined by the orientation

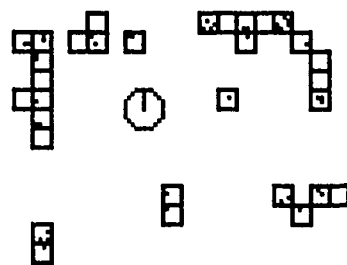


Figure 2. Occupancy grid imposed on the groundplane projection of the vertical stereo features from Figure 1. The grid itself is used to represent the room.

of the robot when the first stereo pair is taken.

4 Recognition

We wish to build the models of locations as the robot explores its world instead of supplying these models *a priori*. The models will therefore be very similar to the data that we obtain. For the first recognition example shown below, the room representation shown in Figure 2 is used as the model. The representation obtained by the robot from another position and orientation in the same room serves as the data in the same recognition example. (The simple model shown here is only the initial model for the room. This room model is updated with the results from each recognition in which it is used (see Sections 5 and 6).)

To recognize a data set with respect to a model, we use a least-squares algorithm to find the best fit of model to data and then evaluate that fit. This fitting process requires a good initial estimate of the transform (2-D translation and rotation) required to match model and data. To obtain these initial estimates, we find transforms that align model and data feature clusters.

The feature clusters are used to obtain the initial alignments consist of colinear groups of model and data points. Cluster pairs from the model are matched with cluster pairs in the data to determine the initial alignment transformations. (Other methods of grouping the points of the representations and aligning model and data will need to be added for other environments [Jacobs, 1988], but these linear groups suffice for our indoor scenes.)

For the initial alignments, a least squares process minimizes the error between the locations of the model and data points by varying the translation (x, y) and rotation θ of the model. This process is similar to the one described by [Lowe, 1987]. However, given an alignment, we find all possible model-data point matches before performing the least-squares optimization instead of adding the pairs as we incrementally refine the transformation. Initially we tried the incremental refinement approach but found that one bad match could affect the rotational component of the transformation enough to generate a completely wrong final result.

The quality of the final recognition is determined by the number of model points that have been matched and

□

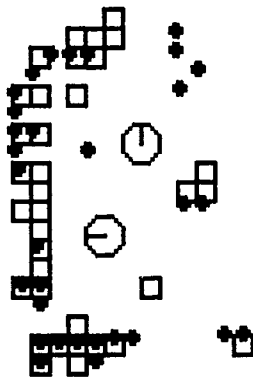


Figure 3: Final recognition using the model from Figure 2 (shown as solid dots) and a new set of data (shown as open squares).

the transform variance. The best recognition obtained from the set of initial guesses is the one that matches the most model points. If more than one recognition matches this same highest number of model points, then the one with the lowest transform variance is chosen (Figure 3).

After finding the best match of model to data, we still need to determine if the match is correct, i.e., if the model and data actually correspond to the same location. We take as the correct recognition the model that matches the highest number of data points above a threshold. Since it is possible that the data does not correspond to a model currently in our database, the threshold is used so that we can declare no recognition in cases where the match is poor.

Matching the wrong model to the data (a false positive) is a serious error since the mobile robot uses this information to verify its location in its world map. Deciding that the current data does not match one of the models in the database when the mobile robot is actually in one of the locations represented by a model (a false negative) is less serious. With no recognition (with respect to an existing model), the current data is added to the database as a new room model. Over the lifetime of the robot, we will be able to merge this new model into the existing model for this room. With data sets obtained from three different rooms, we ran a series of recognition tests with evolving models. (The rooms have almost identical dimensions and were specifically chosen to provide a worst-case test for the system.) The recognition performance improved as the models were built up over time (Figure 4), with the false negative rate remaining below 10% for the fifth and subsequent models. In all cases, recognitions that were accepted also determined the correct model-data transformation. (Extensive testing was performed, the details of which are

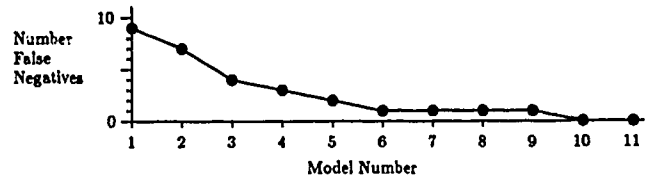


Figure 4: Graph of the number of false negative recognition occurrences versus the model used for recognition. Model 1 incorporates the information from one data set, Model 2 from two data sets, and so on.

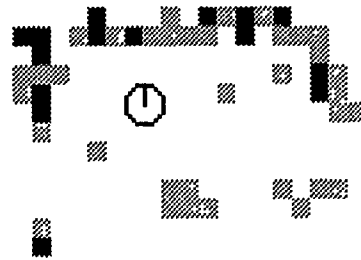


Figure 5: Updated model of Room 914 based on the recognition shown in Figure 3. The darker squares correspond to more heavily weighted model points.

presented in [Braunegg, 1990a].)

5 Model Building

In the recognition above, we used the data obtained from one 360° view of a room for our model. This suffices for an initial model, but a more reliable model can be constructed by combining later views of the same room. Once a model-data recognition has been established, the data is transformed to the same position and orientation as the model and the two combined into a new model. A weight is associated with each point in the model and the weight is increased for those model points that are overlapped by data points (Figure 5). If the current data corresponds to no model in the database, that set of data is entered into the database as the model of a new location.

Combining the new data with the current model serves two purposes. First, new features that appear in the data are added to the model. Second, the features that were used for recognition (and thus had a model-data overlap) become more important in the recognition process. This is accomplished by using a weighted least-squares algorithm to emphasize the matching of heavily weighted model points.

6 Model Maintenance

By combining the model and data sets of a recognition into a weighted model, we build the model up over time (Figure 6). Model points that are seen repeatedly are

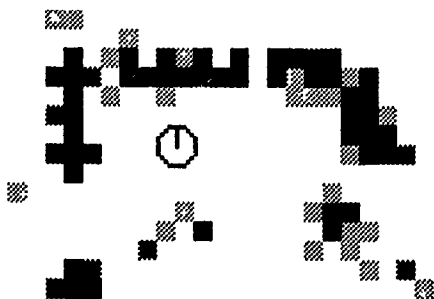


Figure 6: The model for Room 914 after 6 updates.

reinforced in the model, while model points that are not seen again have their weights reduced and are eventually eliminated.

The model update is based on the fact that the more often we see something, the more confident we are of its existence. This is important for three reasons. First, the world changes. By adding newly observed world features into the models, we can incorporate features belonging to objects that are new or are in new locations. By removing those model features that have low weights, we remove from the models those world features that no longer exist or that are no longer in their original locations.

Second, no stereo vision algorithm (or any other sensing scheme) is perfect. There will be both errors of omission and commission. The blind spots will be filled in by later data that is incorporated into the models. The hallucinations will slowly disappear since they will not be seen again and their weights not reinforced.

Third, the locations of world features determined by the stereo algorithm (or, again, any other sensing scheme) are not perfect. When a world feature is seen several times and its location entered into the model, the average perceived location gradually increases in weight. As the data points are added over time, the marked model points approximate a Gaussian distribution about the true location of the feature. Thus, the model updates also serve to refine the locations of the world features in the model. This approach avoids the additional computation needed to model the uncertainty of the sensor data [Durrant-Whyte, 1988] [Matthies and Shafer, 1986].

7 Location Recognition and the World Model

The location (room) models described above are part of a larger world modeling system that is under development. The mobile robot's world is represented by a topological map similar to that described by [Kuipers, 1977]. The world is represented by locations and information about how they are connected. Rather than being supplied *a priori*, this world model is built by the robot as it explores its environment. Because the robot builds and maintains its own world model, it can react to a (gradually) changing world in a way that a robot depending on a precomputed map cannot.

For each location in the world, a model as described previously is built. Since the world locations are dis-

covered through movement (exploration) in the environment, the connections between the locations will be traversed and the location models can be annotated with their positions. Other sensing and robot tasks assign importance to places and features in the environment and these, too, can be annotated on the world model.

8 Conclusions

MARVEL is a working system that builds, maintains, and uses models of world locations for recognition to support navigation. This system uses stereo vision to obtain information about the world for use as input to the model builder and the recognizer. In addition, we have described how this recognition system fits into a larger world modeling scheme.

This work is interesting for several reasons. The model builder and recognition system uses sparse data from the real world as input. We accept the fact that any input data is noisy and explicitly provide a method for handling these data errors. We acknowledge that the world is not static and therefore we provide a way for our location models to change over time as the perceived world changes. We provide for long-term autonomy in our mobile robot by rejecting *a priori* models in favor of models built by the robot itself, thus allowing the robot to explore areas of the world. And finally, we not only build the models that are used by the recognition system, we also maintain them over time in the presence of sensing errors and a changing world.

The recognition system has been tested on over 1000 model-data pairs from actual scenes with less than a 10% false negative recognition rate and a 0% false positive recognition rate. The error rates demonstrated are low enough to permit the inclusion of this world location recognition system in the larger world modeling scheme that we have outlined.

A Models for Room 914

Figures 7 through 18 show the evolving models for Room 914 in the MIT Artificial Intelligence Laboratory. Model 1 was created from Data Set 1. Model 2 is the update of Model 1 based on recognition with Data Set 2. Model 3 is the update of Model 2 based on recognition with Data Set 3, and so on. The weight of a model point is indicated by its darkness. Low weight points are shown as light grey squares while darker squares denote more heavily weighted model points.

References

- [Braunegg, 1989] David J. Braunegg. An alternative to using the 3-D Delaunay tessellation for representing freespace. AI Memo 1185, MIT, September 1989.
- [Braunegg, 1990a] David J. Braunegg. MARVEL: A system for recognizing world locations with stereo vision. Technical Report AI-TR-1229, MIT, May 1990.
- [Braunegg, 1990b] David J. Braunegg. Stereo feature matching in disparity space. In *Proc. IEEE International Conference on Robotics and Automation*, pages 796-803, Cincinnati, OH, May 1990.

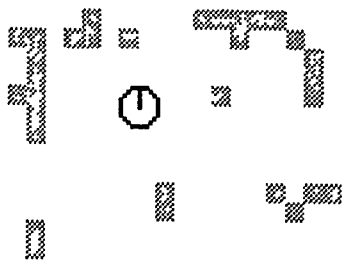


Figure 7: Room 914 Model 1.

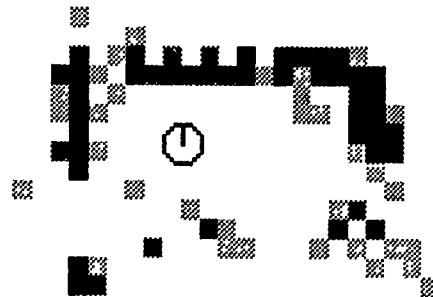


Figure 11: Room 914 Model 5.

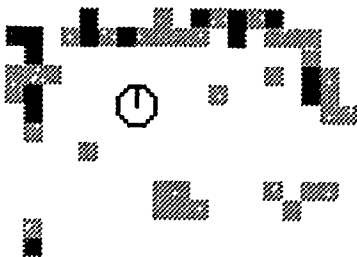


Figure 8: Room 914 Model 2.

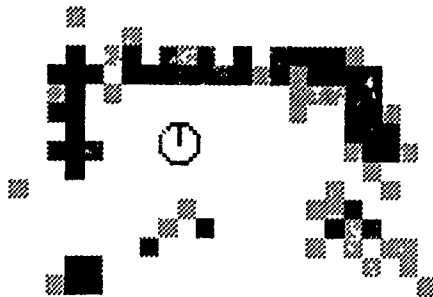


Figure 12: Room 914 Model 6.

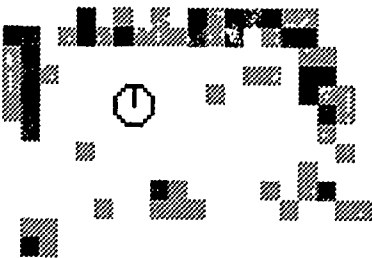


Figure 9: Room 914 Model 3.

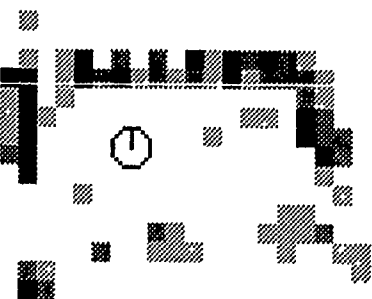


Figure 10: Room 914 Model 4.

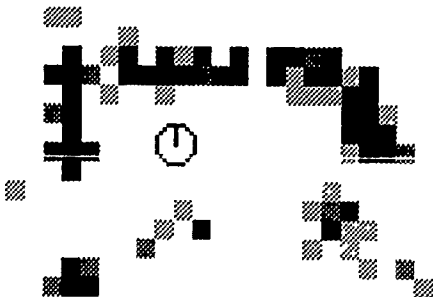


Figure 13: Room 914 Model 7.

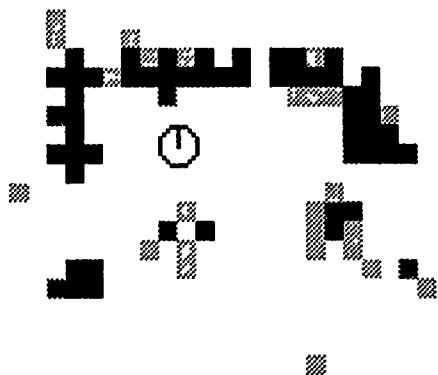


Figure 14: Room 914 Model 8.

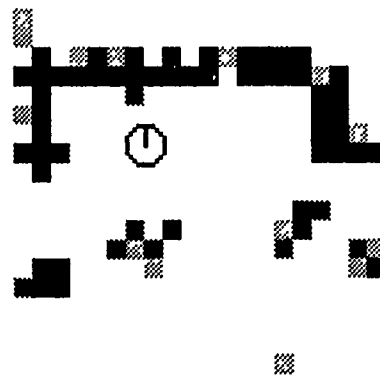


Figure 16: Room 914 Model 10.

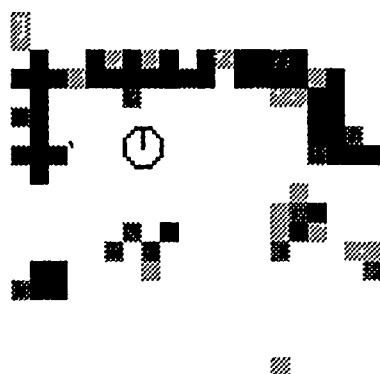


Figure 15: Room 914 Model 9.

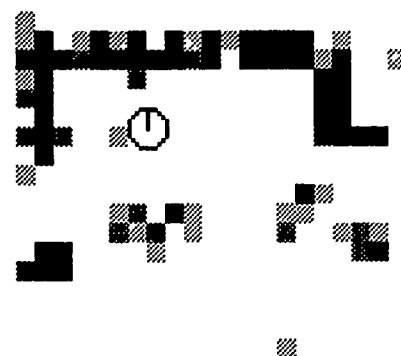


Figure 17: Room 914 Model 11.

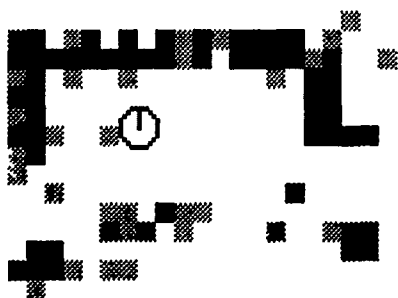


Figure 18: Room 914 Model 12.

[Matthies and Shafer, 1986] L. Matthies and S. A. Shafer. Error modelling in stereo navigation. Technical Report CMU-CS-86-140, Carnegie-Mellon University, 1986.

[Moravec and Elfes, 1985] H. P. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. IEEE International Conference on Robotics and Automation*, pages 116-121, St. Louis, MO, March 1985.

[Brooks, 1985] R. A. Brooks. Visual map making for a mobile robot. In *Proc. IEEE International Conference on Robotics and Automation*, pages 824-829, St. Louis, MO, March 1985.

[Brooks, 1986] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1), April 1986.

[Chatila and Laumond, 1985] R. Chatila and J.-P. Laumond. Position referencing and consistent world modeling for mobile robots. In *Proc. IEEE International Conference on Robotics and Automation*, pages 138-145, St. Louis, MO, March 1985.

[Durrant-Whyte, 1988] H. F. Durrant-Whyte. Sensor models and multisensor integration. *The International Journal of Robotics Research*, 7(6):97-113, December 1988.

[Faugeras *et al.*, 1986] O. D. Faugeras, N. Ayache, and B. Faverjon. Building visual maps by combining noisy stereo measurements. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1433-1438, San Francisco, CA, April 1986.

[Grimson, 1981] W. E. L. Grimson. *From Images to Surfaces: A Computational Study of the Human Early Visual System*. MIT Press, Cambridge, MA, 1981.

[Grimson, 1985] W. E. L. Grimson. Computational experiments with a feature based stereo algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(1):17-34, January 1985.

[Jacobs, 1988] D. W. Jacobs. The use of grouping in visual object recognition. Technical Report AI-TR-1023, MIT, October 1988.

[Kuipers, 1977] B. J. Kuipers. Representing knowledge of large-scale space. Technical Report AI-TR-418, MIT, July 1977.

[Lowe, 1987] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355-395, 1987.

[Marr and Poggio, 1979] D. Marr and T. Poggio. A theory of human stereo vision. *Proceedings of the Royal Society of London*, B(204):301-328, 1979.

TOSS - A System for Efficient Three Dimensional Object Recognition*

Fridtjof Stein and Gérard Medioni
Institute for Robotics and Intelligent Systems
Powell Hall 204
University of Southern California
Los Angeles, California 90089-0273
Email: stein@iris.usc.edu

Abstract

We present an approach for the recognition of multiple three dimensional object models from three dimensional scene data. We are addressing the problem in a realistic environment: the viewpoint is arbitrary, the objects vary widely in complexity, and we make no assumptions about the structure of the surface. We come up with a data structure which we call a *splash*. A splash consists of circular groupings of surface normals. Such a splash captures structural surface properties in a way that we can represent them by sets of two dimensional structures called super segments. Encoded super segments provide the mechanism for fast matching. The acquisition of the three dimensional models is performed automatically by computing splashes in highly structured areas of the objects. For every model all splashes are mapped on super segments. The encoded super segments are recorded in a data base. The scene is screened for highly structured areas. In these areas splashes are computed and mapped on super segments. The encoded super segments retrieve hypotheses from the data base. Clusters of mutually consistent hypotheses represent instances of models. The location of the instance in the scene is found by applying a least squares match on all corresponding points. We present results with our current system TOSS (Three dimensional Object recognition based on Super Segments) and discuss further extensions.

1 Introduction

In this paper we present an object recognition system which is able to match general three dimensional objects in an efficient way. By using the words "three dimensional" we talk about models and scenes having a three dimensional representation. By talking about "general objects" we do not make any assumptions about the shape of the objects. Matching and recognizing in an "efficient way" is based on a fast indexing and retrieval system that has a complexity which grows as $O(kN)$ when N is the number of models, and $k < 1$.

Representing a three dimensional object is either possible by using a surface or a volumetric description. Volumetric descriptions from a single view require a difficult inference step to compensate for the unseen part, so we will use descriptions based on visible surface instead. The task of object recognition involves identifying a correspondence between a part of one range image and a

part of another range image with a particular view of a known object. This requires the ability to match one surface patch of one range image against a surface patch of another range image. The question is: "How can we represent a surface patch so that it can be matched in an efficient way?"

Reviewing the systems of the past, no system (known to the authors) was able to represent, match, and recognize *general* three dimensional objects. Most object recognition systems to date either rely on exact, CAD-like models, or make restrictive assumptions on the possible shape of the surface patches.

1.1 Previous Work

Grimson and Lozano Pérez [8, 9] describe a system which is able to recognize objects from sparse scene data. They exploit geometric constraints to prune the search tree of all possible matches between scene data and model data. Still, the number of combinations that need to be tested grows rapidly with object complexity. If a consistent transformation is found, the object is recognized.

Bhanu [1] presents a 3D scene analysis system for the shape matching of real world 3D objects. Object models are constructed using multiple-view range images. The object is represented as a set of planar faces approximated by polygons. Shape matching is performed by matching the face description of an unknown view with the stored model using a relaxation-based scheme called stochastic face labeling.

Horaud and Bolles [2] present the 3DPO system for recognizing and locating 3D parts in range data. The model consists of two parts: an augmented CAD model and a feature classification network. The model objects are represented by a tree-like network such that each feature contains a pointer to each instance in the CAD models. A local-feature-focus method is used for the matching process.

Faugeras and Hebert [7] developed a system to recognize and locate rigid objects in 3D space. Model objects are represented in terms of linear features such as points, lines, and planes. Range images are used as input. At first, possible pairings between model and scene features are established, the transformation is estimated using quaternions. Then, further matches are predicted and verified by the rigidity constraints.

Ikeuchi [10] developed a method for object recognition

*This research was supported in part by DARPA contract F33615-87-C-1436 and an AT&T grant.

in bin-picking tasks. The models consist of surface inertia, surface relationship, surface shape, edge relationship, extended Gaussian image, and surface characteristic distribution. Since this system is mainly designed for the task of bin-picking, only one type of object, which is the same one as in the model, appears in the scene.

Fan [6, 5] presents a system which takes range images as input and automatically produces a symbolic description of the objects in the scene in terms of their visible surface patches. This segmented representation may be viewed as a graph whose nodes capture information about the individual surface patches and whose links represent the relationships between them. The matching of a scene with a model is based on the comparison of the two graphs.

With 3D-POLY Chen and Kak [4] developed a system in which they present a novel approach of organizing the feature data for three dimensional objects. They present a data structure which they call *feature sphere*. The matching and verification step is based on comparing spatial relationships of special feature sets.

The closest work to our approach was done by Radack and Badler [11]. They introduce a new surface representation called *distance profile*. These profiles are used for the matching process. This method reduces the matching of three dimensional surfaces to the matching of two dimensional curves. They use points with high curvature to position the centers of the distance profiles.

Many systems were developed which based on different assumptions about shape, such as polygonal shapes, solids of revolution or generalized cylinders. In contrast, we believe that our proposed system TOSS (Three dimensional Object recognition based on Super Segments) is able to recognize rigid objects, whose shapes are not constrained by any simplifying assumptions. Our algorithm uses a representation, which is designed to capture the structure (curvature) of a surface patch and allows fast matching.

1.2 Our Previous Work

This paper describes a continuation of our early work [12] which addresses the problem of recognition of multiple flat objects in a cluttered environment from an arbitrary viewpoint (weak perspective). The models are acquired automatically and initially approximated by polygons with multiple line tolerances for robustness. Groups of consecutive linear segments (super segments) are then quantized with a Gray code and entered into a hash table. This provides the essential mechanism for indexing and fast retrieval. Once the data base of all models is built, the recognition proceeds by segmenting the scene into a polygonal approximation; the Gray code for each super segment retrieves model hypotheses from the hash table. Hypotheses are clustered if they are mutually consistent, and represent the instance of a model. Finally, the estimate of the transformation is refined. This methodology allows us to recognize models in the presence of noise, occlusion, scale, rotation, translation and weak perspective. Unlike most of the current systems, its complexity grows as $O(kN)$ when N is the number of models, and $k \ll 1$.

1.3 Plan

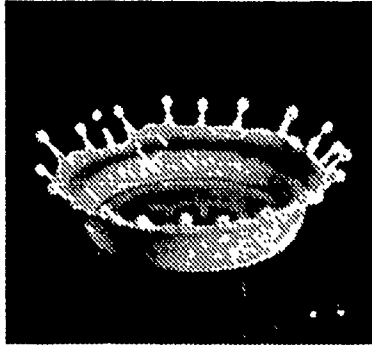
The paper is organized as follows: Section 2 introduces our basic feature, the *splash*. It consists of a group of surface normals which are mapped on a two dimensional structure. We show how we can use our two dimensional approach to represent these two dimensional structures in a way that allows us to match the three dimensional splashes. Section 3 focuses on the representation of a general three dimensional object, the matching and the verification process. In Section 4 we show results of our current implementation.

2 The Splash

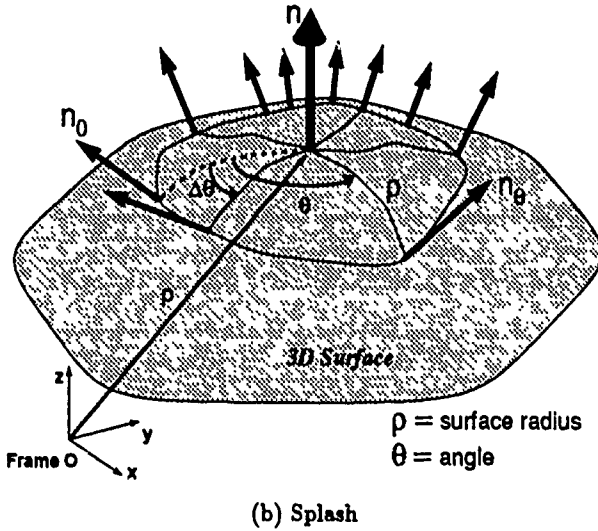
Extending the two dimensional basic feature of the super segment to three dimensions to obtain a feature which represents surfaces is awkward: the polygonal approximation of a two dimensional boundary has a property which is crucial for the super segment idea, but which is not extendable to higher dimensions: the well defined order of the neighborhood of a linear segment. Every segment on a two dimensional polygon has two adjacent neighbor segments. Based on this fact, super segments can be generated by grouping adjacent segments together. In three dimensions this ordered neighborhood property does not exist. Linear or other segmentations of a surface (or volume) lead in general to patches which can have any number and order of neighbor patches. This is a reason why we decided not to go the path of a linear (or higher order) surface segmentation to obtain a representation for matching and recognition. What are the requirements that a representation for general three dimensional objects has to meet? We want the representation to be

1. translation invariant,
2. rotation invariant,
3. general, in that we do not have to make any assumptions about the shape of the object,
4. local enough, so that we can handle occlusion,
5. robust enough, so that we can handle noise.

In the following we will use lower case to describe vectors (n, p, \dots), and upper case to describe coordinate frames (N, O, \dots). The basic feature for representing a general surface patch is the *splash*. The name originates from the famous picture of Professor Edgerton (MIT), showing a milk drop falling into milk (see Figure 1 (a)). This picture bears a resemblance to the normals in our basic feature. A splash is best described by Figure 1 (b). At a given location p we determine the surface normal n . We call this normal the *reference normal* of a splash. A circular slice around n with the surface radius ρ is computed. Starting at an arbitrary point on this surface circle, a surface normal is determined at every point on the circle. In practical we walk around the reference normal with a $\delta\theta$ angle (typically $1^\circ \leq \delta\theta \leq 15^\circ$) and obtain a set of sample points on the surface circle. The normal at the angle θ is called n_θ . A super splash is composed of splashes with different surface radii ρ_i with $i \in \{1, \dots, m\}$, where m is the number of splashes in a super splash.



(a) Milk Splash



(b) Splash

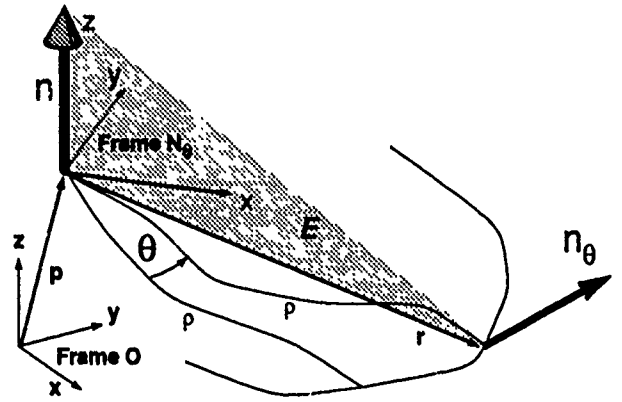
Figure 1: Splashes

We compute a normal in our system by approximating the environment of a normal with triangles of small sizes. Every triangle votes for a triangle normal. The average of the three closest triangle normals is the surface normal. This is a very rough method, but the results were always good enough for our approach.

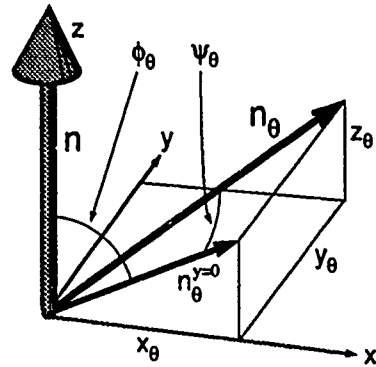
The frame N_θ (see Figure 2 (a)) is defined in the following way:

1. The surface normal n is the z axis.
2. At every location of n_θ , the location of the reference normal p and the tip of the reference normal $n + p$ describe a plane E . The x axis is defined as the vector which is perpendicular to n and lies in the plane E . Furthermore the angle between the z axis and a vector r which is defined between the origin of Frame N_θ and the location of n_θ has to be in the interval $[-90^\circ, 90^\circ]$.
3. The y axis is perpendicular to the x and the z axis in a right handed coordinate system.

This frame has the property that the xy -plane always approximates the tangent plane of the surface in p . We represent n_θ in spherical coordinates: we compute the



(a) Relationship between n and n_θ



(b) Definition of ϕ_θ and ψ_θ

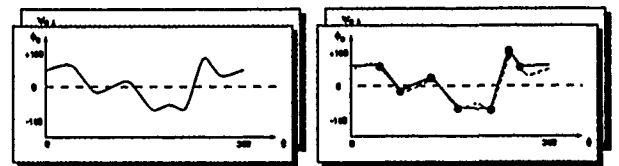
Figure 2: n and n_θ

two angles ϕ_θ and ψ_θ .

$$\phi_\theta = \text{angle}(n, n_\theta^{y=0})$$

$$\psi_\theta = \text{angle}(n_\theta^{y=0}, n_\theta)$$

For every sample point of a splash we obtain such a tuple. Drawing a mapping for ϕ and ψ with respect to θ results in two mappings as in Figure 3(a). These two mappings



(a) ϕ and ψ Mapping with respect to θ

(b) Polygonal Approximation of the ϕ and ψ Mappings

Figure 3: Mapping Tuple

have the following properties:

1. Dependent on where n_0 is, the mappings are shifted along the θ axis.

2. The variation of the curve represents the structural change in the surface environment around the reference normal n .
 - (a) For a splash on a sphere or plane, the mappings are constant.
 - (b) A highly creased surface results in a curved set of mappings.
3. Splashes which are located close to each other have a similar shaped set of mappings. By using the word *similar* we mean similarity in the sense, so that a human would classify them as "pretty much the same". That does not automatically implicate that the pairwise difference results in small values. To be able to compare two mappings, we therefore need a metric.

At this point we have reduced the original question "How do we capture the shape of a general surface patch into a representation?" which is a three dimensional problem, into a two dimensional question "How do we capture the shape of two mappings into a representation?".

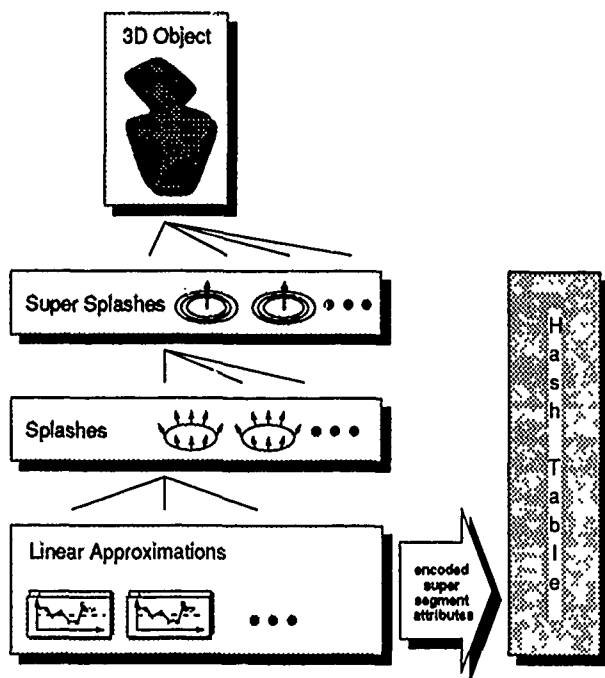


Figure 4: Representation of a Model

3 Recognition

3.1 Object Representation

The solution is straightforward based on our two dimensional approach [12].

1. For all splashes of a model we compute the mappings. In Section 3.4, we talk about the locations of the splashes.
2. For each splash the two mappings are approximated by polygonal approximations (see Figure 3(b)). It is important to note that the mappings (and therefore

the polygons) are periodic ($0^\circ = 360^\circ$). For the purpose of robustness we use multiple line fitting tolerances. Therefore we get a set of polygons for every mapping.

3. For every polygonal approximation we compute a super segment. The start of the super segment is defined at its global maximum value. If there is more than one global maximum we use one super segment for each of the maxima. With this super segment choice, we obtain rotational invariance in our representation. By starting all super segments at the maximum of the approximation, two shifted polygons with the same shape result in the same super segment.
4. All the obtained super segments are encoded. The encoding works as described in [12]. As encodable attributes we take
 - (a) the angles between two consecutive segments of a super segment (they capture the curvature information)
 - (b) the mapping label ϕ or ψ
 - (c) the maximum of the mapping (ϕ_{max} or ψ_{max})
 - (d) the surface radius of the splash.

Incorporated in the code of the angles of the super segments is also the cardinality (number of segments) of the super segments (by the number of angles). That avoids matching super segments of different cardinality. The encoding of ϕ_{max} or ψ_{max} allows to distinguish between different curved surfaces of the same shape (e.g. two spherical surfaces with different sphere radii). The encoding of the radius avoids matches between splashes with different splash radii.

5. All the encoded super segments serve as keys into a hash table (the data base), where we record the corresponding splashes as entries (see Figure 4).

3.2 Matching

By using indexing for the matching process, we only select a small set of candidate models that are likely to be present in the scene. We assume that most objects in our data base (hash table) are redundantly specified by their splashes. The scene is preprocessed as explained in Section 3.1 to generate all the splashes and their super segments. The encoded super segments are used to retrieve the matching hypotheses between the splashes of model and scene.

3.3 Verification

The verification stage is fully described in [12] and consists of the following steps:

1. We compute all possible matches for the splashes of the scene with the model splashes to generate multiple hypotheses.
2. These hypotheses are stored with respect to the model they vote for.

3. The next step is the formation of consistent clusters based on angle and distance constraints. A cluster of mutually consistent hypotheses represents an instance of a model.
4. After this grouping of hypotheses into clusters, we can compute the transformation from the model coordinates to the scene coordinates by applying a least squares calculation on all the matching splashes. Because of noise, we get in general a good first guess for the transformation but not an exact match. A second least squares match on corresponding corners or segments can refine the result.

3.4 Interest Operator

One question remains open: at which locations of an object should we compute the splashes? The brute force answer would be: at every pixel (in a range image). A more sophisticated answer would include the observation that we will not get *structurally rich* splashes at every point, which lead to good and unambiguous matches. Splashes in flat areas result in super segments with low cardinality. Super segments with low cardinality are less descriptive than super segments with high cardinality, which represent high structured surface patches. Therefore to obtain good and unique matches we are interested in matches of structured patches and high cardinality. These can be found at or near points of high curvature.

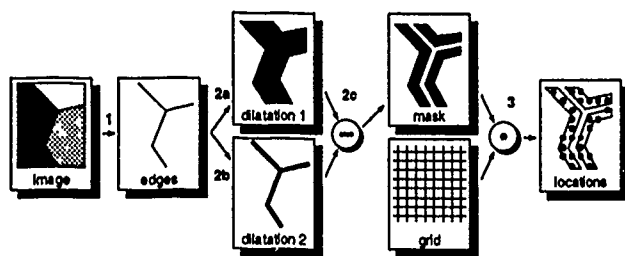


Figure 5: Interest Operator

Our simple selection method works as follows (see Figure 5):

1. We compute the edges of the artificial shaded range image (by assuming a light source at the viewer and computing a gray value for every pixel in the range image under the assumption of Lambertian conditions) with the Canny edge detector [3]. We want to position the splashes in areas where we can expect structured patches on one object. This property is not given on the boundary. A boundary edge typically has the object as one neighborhood and other objects or background information as the other neighborhood. Therefore we use only the "inner object edges" and throw away the boundary edges.
2. For positioning the splashes we are interested in areas around the edges. Placing a splash on a high curvature point has the disadvantage of an unreliable reference normal. A reliable reference normal is important for a stable splash. Nevertheless we

want to capture the structure of the edges in the splash. Therefore the best place for a splash is in the neighborhood of an edge. We get this area in 3 steps:

- (a) We dilate the edge image by replacing every pixel on the edges by a disc of a certain radius (e.g. $r_1 = 8$ pixels). The resulting image is called *dilatation 1*.
- (b) We dilate the edge image with another radius (e.g. $r_2 = 3$ pixels with $r_1 > r_2$). The resulting image is called *dilatation 2*.
- (c) The subtraction of dilatation 1 and dilatation 2 gives us a mask. This mask describes an area with the above described characteristics. Points in this mask are no high curvature points, but they are close to edges.

3. We compute a grid of splashes on the range image with respect to this mask.

As we will see in Section 4, this simple method works pretty well.

3.5 Complexity Analysis

In [12] we show that for the two dimensional case, under the assumptions that every model has the same number of super segments and that the entries are equally distributed over the hash table, the overall cost is

$$O_{recognize} = O_{match} + O_{verify} = O(q) + O(M) = O(M),$$

where M is the number of models in the data base and q the number of super segments in the scene. We assume q constant to study the behavior of a large data base (large M). We show further that the slope of the linear cost function is dependent on the occurrence of a model in the scene or its absence. In our results for the two dimensional case the cost for detecting the absence of a model is less than 10% of the cost for the detection of the occurrence. In the three dimensional case, we map each splash onto a constant number of super segments. Therefore we claim that this result is also valid for the three dimensional case. To support this claim, we created a data base of 100 objects. Every object consists of a random range image. The scene is a composition of four of these objects including translation, rotation, and occlusion. In our results the cost for detecting the absence of a model is less than 50% of the cost for the detection of the occurrence. We believe that the cause for the higher relative cost of absence detection in three dimensions compared to two dimensions lies in the fact that splashes for surfaces are less descriptive than super segments for boundaries.

4 Results

With the current implementation we are able to show that the proposed recognition mechanism of recognizing general three dimensional objects works. We choose two different scenes:

1. A Mozart bust, which is highly curved, and which partially has a structured surface. Because of lack of data we cannot deal with a real three dimensional

model and a scene which consists of range data. Therefore we take the original data of the Mozart bust as model, rotate the range data synthetically to obtain the scene. We rotate pixel by pixel and fill the holes by averaging the values of neighbor pixels. This rotation process is guaranteed to add a lot of noise!

2. A scene composed of a plane and a wagon, which shows that our method works for objects which can be approximated by polygonal surfaces. We have four range images, two of the plane from different views and two of the wagon from different views. One wagon and one plane image serve as models. The scene is composed synthetically by combining the other two range images into the scene image.

4.1 Mozart

Our input data is the range image. For better visibility we show the artificially shaded images. Figure 6(a) shows the model of the Mozart bust, Figure 6(d) shows the scene, which is the model rotated by 20 degrees around a tilted axis. The inner edges are shown in Figure 6(b) and (e). The results of our interest operator are the masks shown in Figure 6(c) and (f). The recognition result is shown in Figure 7. (We overlayed a grid of the range image of the model, transformed by the resulting transformation on top of the Figure 6(d).)

4.2 Plane and Wagon

Figure 8(a) shows the shaded range image of the plane and Figure 8(b) shows the shaded range image of the wagon. Figure 9 shows the best detected solution. It is interesting to note that the plane has no highly structured areas on the wings. Therefore we get no matches on the wings. This leads to poor correspondence for the wings (we have one degree of freedom along the axis of the plane).

object (size in pixels)	# super splashes	radii (pixels)	tolerances (degrees)	grid (pixels)
mozart (264x399)	402	20,30,40	15,20,25,30	6
scene 1 (240x390)	329	20,30,40	15,20,25,30	8
plane (507x218)	60	20,30,40	15,20,25,30	8
wagon (422x178)	367	20,30,40	15,20,25,30	6
scene 2 (458x324)	293	20,30,40	15,20,25,30	8

Table 1: Table of Objects

We can give some rough numbers about the time complexity (on a serial Symbolics machine).

1. The acquisition of one super splash (consisting of typically 3 splashes with 4 line fitting tolerances) takes about 7 seconds.
2. The Mozart bust consists of about 400 super splashes, therefore it took about 45 minutes to compute all splashes.

3. The plane consists of about 60 splashes, therefore it took about 7 minutes to compute all splashes.
4. The matching process is always below 20 seconds.
5. The verification process takes less than 3 minutes.

The items 1 to 3 are processed offline to build the data base. The recognition process itself consists of item 4 and 5. All these numbers reflect neither the high parallelism which is theoretically possible nor the data redundancy with which we work at the moment. Simple improvements can significantly increase the performance. This is the goal of our future studies.

5 Conclusion and Future Research

The results with our current implementation of the TOSS system described in the last sections show that the idea of describing the surface of an object based on splashes is powerful enough to handle complex three dimensional shapes. Our future research will extend various aspects of this mechanism. Our system is designed for the recognition of surface patches. Several other pieces of information are not used to enhance the performance. One is for example the boundary of an object. There might be a possibility of including the different boundaries derived from different views of the object by including two dimensional super segment recognition in the three dimensional recognition process. We have to study this possibility. Our long term goal is to build a recognition system which is able to recognize three dimensional models in a two dimensional gray level image.

References

- [1] B. Bhanu. Representation and Shape Matching of 3-D Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(3):340-350, 1984.
- [2] R. C. Bolles and P. Horaud. 3DPO: A Three-Dimensional Part Orientation System. *International Journal of Robotics Research*, 5(3):3-26, 1986.
- [3] J.F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 8(6):679-698, November 1986.
- [4] C. H. Chen and A. C. Kak. A Robot Vision System for Recognizing 3-D Objects in Low-Order Polynomial Time. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1535-1563, 1989.
- [5] T. J. Fan. *Describing and Recognizing 3-D Objects Using Surface Properties*. Springer Verlag, New York, 1990.
- [6] T. J. Fan, G. Medioni, and R. Nevatia. Recognizing 3-D Objects Using Surface Descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1140-1157, 1989.
- [7] O. D. Faugeras and Hebert M. The Representation, Recognition, and Locating of 3-D Objects. *International Journal of Robotics Research*, 5(3):27-52, 1986.
- [8] W. E. L. Grimson and T. Lozano-Pérez. Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research*, 3(3):3-35, 1984.

- [9] W. E. L. Grimson and T. Losano-Pérez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):469-482, 1987.
- [10] K. Ikeuchi. Precompiling a geometrical model into an interpretation for object recognition in bin-picking tasks. In *Proceedings of the DARPA Image Understanding Workshop*, pages 321-339, Los Angeles, California, February 1987.
- [11] G. M. Radack and N. I. Badler. Local Matching of Surfaces Using a Boundary-Centered Radial Decomposition. *Journal of Computer Vision, Graphics, and Image Processing*, 45:380-396, 1989.
- [12] F. Stein and G. Medioni. Graycoding and Indexing: Efficient Two Dimensional Object Recognition. In *Proceedings of International Conference on Pattern Recognition*, Atlantic City, New Jersey, June 1990.



Figure 7: Scene 1: Result of Recognition



(a) Mozart: Shaded Range Image (b) Mozart: Inner Edges (c) Mozart: Interest Mask

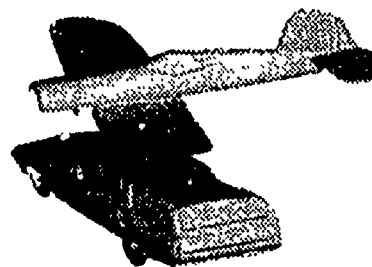


(d) Scene 1 (e) Scene 1: Inner Edges (f) Scene 1: Interest Mask

Figure 6: Mozart Scene



(a) Plane: Shaded Range Image (b) Wagon: Shaded Range Image



(c) Scene 2: Shaded Range Image

Figure 8: Plane and Wagon Scene



Figure 9: Scene 2: Result of Recognition

Recovering Shape from Contour for SHGCs and CGCs *

Fatih Ulupinar and Ramakant Nevatia
Institute for Robotics and Intelligent Systems
School of Engineering
University of Southern California
Los Angeles, California 90089-0273

Abstract

We analyze the properties of Straight Homogeneous Generalized Cones (SHGCs) and Constant Generalized Cylinders (CGCs), and derive the types of symmetries that the limb boundaries and cross sections of these objects produce on the image plane. The constraints on the 3-D shape of the objects are formulated based on the symmetries and from the geometry of the projection models. Finally the methods that recover the 3-D shape from the image of their contours are discussed and recovered surfaces are shown for sample objects.

1 Introduction

This paper is about inferring 3-D shape from 2-D contours for a class of objects, namely generalized cones of constant cross-section (but possibly having complex shaped axes) which we call CGCs (or *snakes*) and for straight homogeneous generalized cones or SHGCs. This class of generalized cones covers a broad class of objects of interest, it includes the so-called linear straight homogeneous generalized cones, solids of revolution, and pipes of arbitrary shape. Some examples are shown in figures 1 and 2. The method we describe is based on, and is a major generalization of, the technique we developed for inferring shape of zero-Gaussian curvature (or ZGC) surfaces [Ulupinar and Nevatia, 1988, Ulupinar and Nevatia, 1990].

Inferring shape of the surfaces in a scene from a single line drawing is an important and difficult problem in computer vision. Early work concentrated on analysis of line drawings of polyhedra [Huffman, 1971, Clowes, 1971, Mackworth, 1973, Kanade, 1981, Sugihara, 1986]. There have been other efforts at developing techniques for curved surfaces such as [Barrow and Tenenbaum, 1981, Stevens, 1981, Xu and Tsuji, 1987, Horaud and Brady, 1988]. We believe that the techniques presented here extend the complexity of surfaces that can be analyzed significantly.

*This research was supported by the Defense Advanced Research Projects Agency under contract number F 33615-87-C-1436 monitored by the Air Force Wright Aeronautical Laboratories, Dayton, Ohio Order No. 3119.

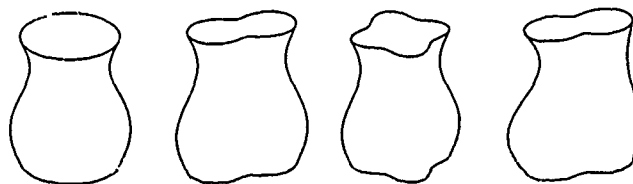


Figure 1: Sample SHGCs.

Our approach is based on an analysis of the symmetries in a scene. In section 2 we define the symmetries we use. Then we show how such symmetries arise naturally in images of the class of objects we study. In section 3 we summarize the constraints that derive from these symmetries and other properties of boundaries for determining the 3-D shape. In section 4 we give a summary of previous work on ZGC surfaces. In sections 5 and 6 we show how these constraints, and other properties of the boundary allow us to infer 3-D shape of the objects in the scene. Some computational results are also shown.

In the subsequent analysis, we will assume that the image is obtained by an orthographic projection (though some of our theorems apply to perspective projection as well) and from a general viewpoint.

Definition 1 General Viewpoint : *A scene is said to be imaged from a general viewpoint, if perceptual properties of the image are preserved under slight variations of the viewing direction.*

Specifically, the properties we are interested in are: straightness and parallelity of lines and symmetry of curves (symmetries as defined in the following).

2 Symmetry Definitions and Qualitative Shape Inference

We believe that symmetries have an important role in shape perception, this also has been noted and used by many researchers [Nevatia and Binford, 1977, Nalwa, 1987, Rao, 1988, Kanade, 1981, Stevens, 1981]. We first define two types of symmetries and then show the conditions under which they may be observed in an image of CGC or SHGC objects.

2.1 Symmetry Definitions

We define two types of symmetries, that we call *parallel symmetry* and *mirror symmetry*. For curves to be

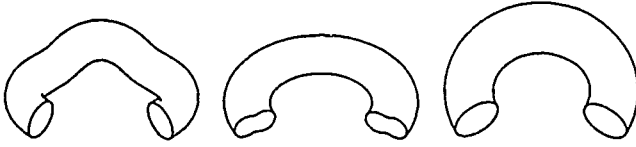


Figure 2: Some sample PRCGCs

symmetric (parallel or mirror) certain point-wise correspondences between two curves must exist. We will call the lines joining the corresponding points on the curves as the *lines of symmetry*, the locus of the mid points of these lines as the *axis of symmetry*, and the curves forming the symmetry as the *curves of symmetry*.

Parallel Symmetry Let $X_i(s) = (x_i(s), y_i(s), z_i(s))$, for $i = 1, 2$, be two curves in 3-D parameterized by arc length s .

The curves $X_1(s)$ and $X_2(s)$ are said to be parallel symmetric if there exists a point-wise correspondence $f(s)$ between them such that, $X_1'(s) = X_2'(f(s))$ for all values of s for which X_1 and X_2 are defined and $f(s)$ is a continuous monotonic function. Note that projection of curves X_1 and X_2 under orthographic projection produces image curves that are parallel symmetric such that the 3-D point correspondence is preserved. Computing symmetry between two curves using this definition requires estimating the function $f(s)$ as well. A useful special case is when $f(s)$ is restricted to be a linear function.

Mirror Symmetry For mirror symmetry, the point-wise correspondence should be such that the axis of the symmetry is straight, and the lines of symmetry are at a constant angle (not necessarily orthogonal) to the axis of symmetry. This definition of the mirror symmetry is similar to that of skew symmetry. We use the term mirror symmetry in the context of curved surfaces as skew symmetry has historically been used for planar surfaces only.

We believe that the symmetries we have defined, either separately or taken together, give some qualitative as well as quantitative information about the surface shape. In [Ulupinar and Nevatia, 1990] we showed that a figure bounded entirely by one mirror symmetry must be planar and that a figure bounded by one parallel symmetry and one mirror symmetry with straight lines of symmetry must be a ZGC surface (assuming general viewpoint in both cases). In the following we show the properties that allow us to infer the presence of PRCGCs and SHGCs.

First, we discuss some useful geometric properties of differentiable surfaces.

2.2 Surfaces and Their Limb Edges

Definition 2 Tangent line, L_v , of a surface, S , at point, P , in a given direction, V , is the line from the point P in the direction of the tangent of the curve, C , obtained by cutting the surface by a plane, Π , that passes through P , and contains the normal, N , of the surface at P and the direction given by the vector V .

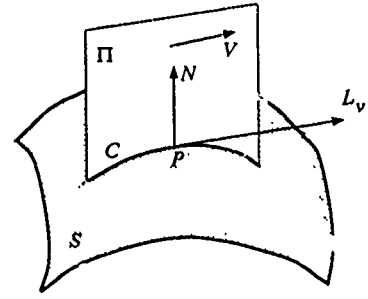


Figure 3: Tangent line, L_v , of a surface S at point P in direction V .

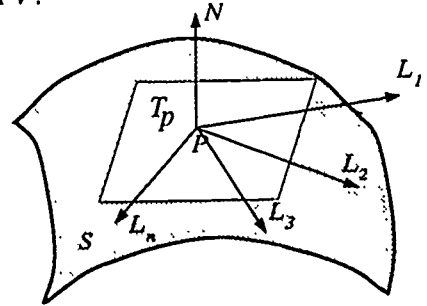


Figure 4: Tangent plane, T_p , of a surface, S , containing all the tangent lines at point P

Figure 3 shows an example.

It is a well known property in differential geometry [Do Carmo, 1976] that the tangent lines, L_{v_i} , of a surface, S , at point, P , in all possible directions, $V_i \in R^3$, are on a plane, T_p , called the tangent plane of the surface at P . Moreover the plane T_p is orthogonal to the normal, N , of the surface at P . This property is shown graphically in figure 4.

Next, we define limb edges and their projections for smooth surfaces.

Definition 3 The limb edge of a surface is a viewpoint dependent curve on the surface such that at each point on the curve the surface normal is orthogonal to the viewing direction.

The limb edges project on the image plane as the bounding curve of the surface. At these edges the surface smoothly curves around to occlude itself. This definition of limb edges holds both for orthographic and perspective projection. Limb edges (also called "occluding contours") can give some very important information about the 3-D surface they come from; Koenderink [Koenderink, 1984] has given a nice analysis in previous work. We will show how the limb edges help us recover 3-D surface shape later in this paper.

Theorem 1 All the tangent lines of a surface at a point, P , which is on a limb edge of the surface for a given projection geometry, project as the same line on the image plane.

Proof The proof involves a simple combination of the definition of limb edges and the property of tangent planes. Since the normal of the tangent plane at P

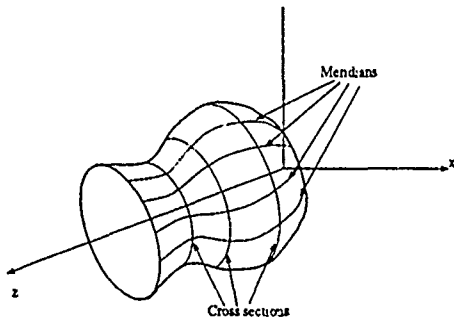


Figure 5: An SHGC along the z coordinate axis with both meridians and cross sections marked.

(which is also the normal of the surface at P) is orthogonal to the viewing direction, the tangent plane projects as a line on the image plane. Therefore all the tangent lines at P , which are included in the tangent plane also project to the one line that the plane projects into. \square

This theorem, though simple and rather obvious, turns out to be highly useful in proving other important properties of limb boundaries.

2.3 SHGCs

Straight homogeneous generalized cones (SHGCs) are obtained by sliding a cross section, say C , along a straight axis, say A . The cross section is also scaled as it is swept along the axis by a scaling function, say r . We can parameterize the surface, S , of an SHGC, given the planar cross section $C(u) = (x(u), y(u), 0)$, and the scaling function $r(t)$, as :

$$S(u, t) = (r(t)x(u), r(t)y(u), t) \quad (1)$$

The axis of the SHGC in this case is the z axis of the coordinate system. An example is shown in figure 5. Note that the cross section curves are generated by fixing t and varying u . We will call the curves generated by fixing u and varying t as the meridians of the surface. Note that cross section of an SHGC are planar because the cross section function $C(u)$ is planar, and the meridians of an SHGC are planar since the SHGC has no twist in its sweep. Let meridian edges of an SHGC be edges that are along the meridians of the SHGC. Usually images of SHGCs do not contain meridian edges, however, such edges may be present if the cross section has a tangent discontinuity (a corner). Figure 1 shows some sample SHGCs.

Theorem 2 For an SHGC, the tangent lines of the surface in the direction of the axis from the points of any given cross section intersect at a common point on the axis of the SHGC.

A proof of this theorem may be found in [Shafer and Kanade, 1983]. Figure 6 (a) graphically illustrates the property.

Corollary The tangents of all meridian edges at the points they intersect a single cross section intersect the axis of the SHGC at a single point. Therefore in the image plane, too, the tangents of the images of the meridian edges, at the point they intersect a single cross

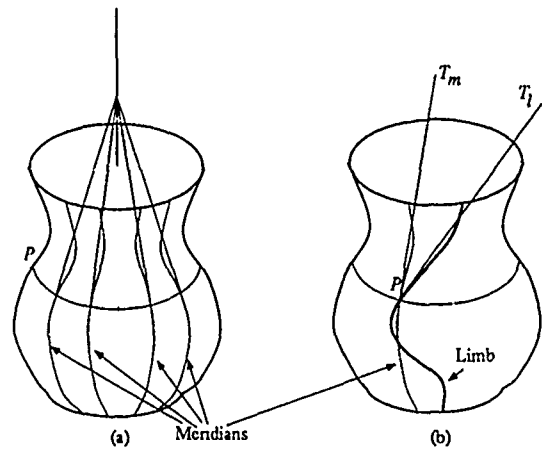


Figure 6: (a) An SHGC, and its tangent lines, in the direction of the axis emitting from a single cross section, intersecting at a single point on the axis. (b) The tangent lines, T_l , of limb edges are not the same as the tangents lines, T_m , of the meridians in 3-D.

section, intersect the image of the axis in a single point, under orthographic or perspective projection.

It has been shown by Shafer[Shafer, 1983] that the limb edges on an SHGC are not planar. Therefore the limb edges of an SHGC are necessarily not along its meridians, and the tangents of the limb boundaries at the point they intersect the same cross section do not intersect the axis in 3-D. (Figure 6 (b) shows the limb edge and its tangent for an SHGC after rotating it, to show that in 3-D the tangent of the limb edge does not intersect the axis of the SHGC.) Still, it has been shown by Ponce [Ponce *et al.*, 1989] that under orthographic projection the tangents of the limb edges, at the point they intersect the same cross section, intersect the image of the axis at a single point. Here we give a simpler proof which is independent of the projection geometry.

Theorem 3 The tangents of the projections of the limb edges at the points they intersect the same cross section, when extended, intersect the image of the axis of the SHGC at the same point.

Proof Say the limb edge intersects a given cross section at point P (see figure 6). Since the tangent line T_m from point P in the direction of the axis of the SHGC (the tangent line of the meridian passing through the point P) intersect the axis of the SHGC, by theorem 1, the image of the tangent line T_l from point P in the direction of the tangent of the limb edge project as the same line as the tangent line T_m and thus image of the line T_l intersect the image of the axis at the same point as the image of the line T_m intersects. \square

Since theorem 1 holds both under perspective and orthographic projection, the above theorem and the proof hold for both of the projection geometries.

In the following we show that the cross sections of an SHGC are parallel symmetric in 3-D with the meridian curves joining the parallel symmetric points of the cross sections.

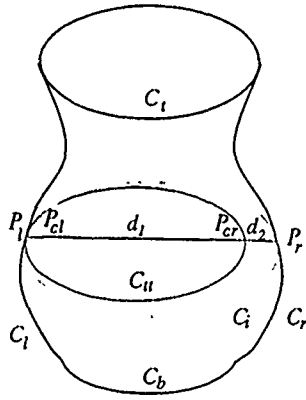


Figure 7: Image of an SHGC cut along its cross sections. Image of the top cross section curve is C_t , the bottom one is C_b and the limb boundaries are on the left C_l and on the right C_r .

Theorem 4 *The cross sections of an SHGC are parallel symmetric in 3-D with each other such that the meridian curves join the parallel symmetric points of the cross sections.*

Proof We have to show that the direction of the tangent of the cross sections is independent of the t parameter curve. Using the parameterization for an SHGC given in equation 1 the tangent of the cross sections (u parameter curves) is given by:

$$S_u = (r(t)x'(u), r(t)y'(u), 0) = r(t)(x'(u), y'(u), 0) \quad (2)$$

Clearly the direction of S_u is independent of the t parameter. \square

Corollary The projection of the cross section curves of an SHGC are also parallel symmetric in the image plane. And the correspondence function is linear because cross sections are obtained by scaling a reference cross section curve without deforming it.

2.3.1 Recovering the Cross sections

We next show how to find the projections of cross sections in the image of an SHGC, given the images of its external contours. Our method does not require complete cross sections, but only the part that lies on the visible face of the SHGC. However, we require that the SHGC be cut along its cross sections, otherwise we would not have a parallel symmetry between the image curves of the two extreme cross-sections (C_t and C_b in figure 7). We conjecture that humans too do not do well if this condition is not satisfied. The following algorithm recovers the image curves C_i that correspond to the projections of the cross sections of the SHGC.

For each point $P_l \in C_l$ do:

1. Find the point $P_{cl} \in C_t$ such that $C'_t(P_l) \equiv C'_t(P_{cl})$.¹

¹The \equiv operator is used for parallelity of vectors, that is, if $V_1 \equiv V_2$ then $V_1 = \lambda V_2$ for some scalar λ .

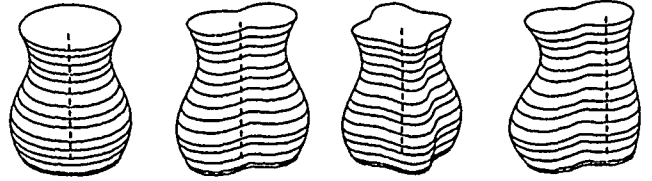


Figure 8: Images of the cross sections and axes recovered for the SHGCs in figure 1

2. Translate the cross section curve C_t such that the point $P_{cl} \in C_t$ coincides with the point P_l , obtaining the curve C_{tl} .
3. Find the point $P_{cr} \in C_{tl}$ that minimizes the function $f(P_{cr}) = (d_1 + d_2)/d_1$ which is the amount of scaling required to be applied on the curve C_{tl} to bring the point P_{cr} to the point P_r . The quantities d_1 and d_2 are the length of the line segments from P_l to P_{cr} and from P_{cr} to P_r . It can be shown that local minima of the function $f(\cdot)$ above gives the correct point $P_{cr} \in C_{tl}$ such that the limb boundary condition $C'_{tl}(P_{cr}) \equiv C'_r(P_r)$ is met.
4. Scale the curve C_{tl} by $f(P_{cr})$ so that the point P_{cr} meets with the point P_r , obtaining the curve C_i .

The curve C_i obtained by this algorithm is precisely the image of the cross section curve between the points P_l and P_r of the SHGC. Once the correspondence of the points P_l and P_r between the limb edges C_l and C_r is obtained, we can recover the image of the axis of the SHGC by using theorem 3. Figure 8 shows the computed images of the cross section curves and the axes for SHGCs in figure 1. If the parallel symmetric points of the cross section curves are joined, by theorem 4, we obtain the meridian curves.

2.3.2 Observing SHGCs

If there are two parallel symmetric curves with a linear correspondence function such that they are bound by curves that has a straight axis when the axis is computed by the above algorithm, then we can hypothesize that the line drawing results from an SHGC.

2.4 CGCs (Snakes)

Snakes are generalized cones that have a constant cross section but the axis may be an arbitrary 3-D curve. Following Shafer's terminology [Shafer *et al.*, 1983], such objects may be called CGCs. We will focus on CGCs that have planar axis and that are "right", ie the cross sections are orthogonal to the axis; we call such objects PRCGCs. Figure 2 shows some examples.

In the following, we show that limb boundaries of a PRCGC project as parallel symmetric curves under orthographic projection.

Let us choose a coordinate system such that the axis of the PRCGC lies in the $x - z$ plane and one of the cross-sections, say $C(u) = (c_x(u), c_y(u), 0)$, is aligned with the $x - y$ plane. Let $A(t) = (a_x(t), 0, a_z(t))$ be the axis parameterized in terms of its arc length, that is, $|A| = a_x^2 + a_z^2 = 1$ for all t . Also, let $A(0) = (0, 0, 0)$ and since the cross section is orthogonal to the axis $A'(0) =$

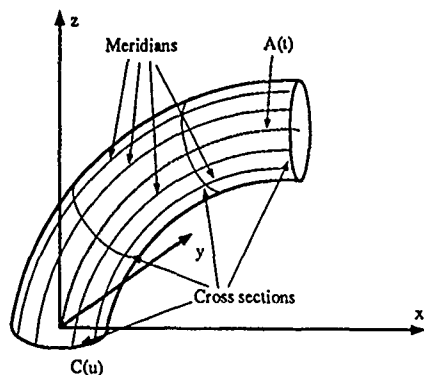


Figure 9: A PRCGC with both meridians and cross sections marked.

(0, 0, 1). Then the surface of the PRCGC, $S(u, t)$ is given by:

$$S(u, t) = R(A'(0), A'(t)) \cdot C(u) + A(t) \quad (3)$$

where $R(V_1, V_2)$ is the rotation matrix that transforms the direction vector V_1 into vector V_2 . For $A'(0) = (0, 0, 1)$ and $A'(t) = (a'_x(t), 0, a'_z(t))$ the rotation matrix R becomes:

$$R = \begin{bmatrix} a'_z(t) & 0 & a'_x(t) \\ 0 & 1 & 0 \\ -a'_x(t) & 0 & a'_z(t) \end{bmatrix} \quad (4)$$

Note that the curves generated by fixing t and varying u are the *cross sections* of the surface $S(u, t)$. We will call the curves generated by fixing u and varying t as the *meridians* of the surface. The meridians are also the loci of points on the cross section as the cross section is swept along the axis. Figure 9 shows an example.

Lemma 1 *The meridians of a PRCGC are parallel symmetric and the curves joining the parallel symmetric points of the meridians form the cross sections of the surface.*

Proof We need to show that the direction of the tangents of the surface in the direction of the meridians, $\frac{\partial S}{\partial t}$, is independent of the parameter u .

$$\begin{aligned} \frac{\partial S(u, t)}{\partial t} &= \frac{dR}{dt} \cdot C(u) + A'(t) \\ &= \begin{bmatrix} a''_z(t) & 0 & a''_x(t) \\ 0 & 0 & 0 \\ -a''_x(t) & 0 & a''_z(t) \end{bmatrix} \cdot \begin{bmatrix} c_x(u) \\ c_y(u) \\ 0 \end{bmatrix} + \begin{bmatrix} a'_x(t) \\ 0 \\ a'_z(t) \end{bmatrix} \\ &= \begin{bmatrix} a'_x(t) \\ 0 \\ a'_z(t) \end{bmatrix} + c_x(u) \begin{bmatrix} a''_z(t) \\ 0 \\ -a''_x(t) \end{bmatrix} \\ &= A'(t) + c_x(u)(A''(t))^\perp \end{aligned} \quad (5)$$

where $(A''(t))^\perp$ is a vector which is orthogonal to the vector $A''(t)$ and is in the $x - z$ plane. Also note that, $A''(t) \cdot A'(t) = 0$ since

$$0 = d(1) = d(A'(t) \cdot A'(t)) = 2A'(t) \cdot A''(t) \quad (6)$$

We conclude that the vector $(A''(t))^\perp$ is parallel to the vector $A'(t)$, since $A'(t) \perp A''(t)$, $A''(t) \perp (A''(t))^\perp$ and all

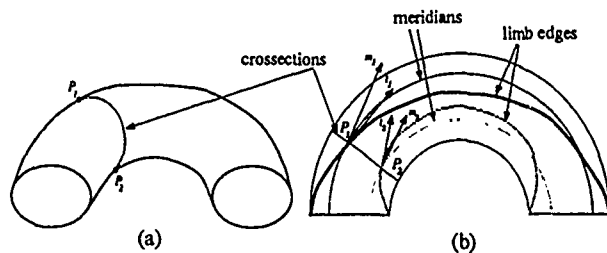


Figure 10: A PRCGC (half of a torus) (a) from a general view and (b) semi-transparent top view with the limb edges of the previous view and the meridians passing from the points P_1 and P_2 marked along with their tangent lines.

three vectors are on a plane (the $x - z$ plane). Then, we can rewrite $\frac{\partial S}{\partial t}$ as:

$$\frac{\partial S(u, t)}{\partial t} = (1 + c_x(u) \frac{|(A''(t))^\perp|}{|A'(t)|}) A'(t) \quad (7)$$

It is obvious that while the length of the vector $\frac{\partial S}{\partial t}$ depends on the u parameter, the direction of it is independent of the u parameter. \square

Although the meridian curves on a PRCGC are parallel symmetric it can be shown that the limb edges of a PRCGC are not necessarily parallel symmetric in 3-D (see Figure 10). However, the following theorem proves that the projections of the limb edges of a PRCGC are parallel symmetric under orthographic projection.

Theorem 5 *The limb edges of a PRCGC project as parallel symmetric curves onto the image plane.*

Proof Here we use the property given in theorem 1 and in lemma 1. Consider the points P_1 and P_2 in figure 10 such that both points are on the same cross section. As can be seen in figure 10 (b) the tangent lines l_1 and l_2 from points P_1 and P_2 in the direction of the limb edges are not parallel symmetric in 3-D. However, the tangent lines m_1 and m_2 from points P_1 and P_2 in the direction of the meridians are parallel symmetric by lemma 1. Since the tangent line l_1 project the same line as the tangent line m_1 , and tangent line l_2 project the same line as the tangent line m_2 by theorem 1 the projection of the limb boundaries of a PRCGC are parallel symmetric. \square

2.4.1 Observing PRCGCs

If in the image plane there are parallel symmetric curves that are terminated by two curves (possibly closed and having mirror symmetry which enhances planarity of the cross section) then we hypothesize that it is a PRCGC. The real test for the line drawing to belong to a PRCGC may be performed after the cross sections are recovered as described in section 6.1.

3 Constraints for Determining Surface Shape

We now give three constraints that derive from observations of the symmetries and other boundaries in the image.

3.1 Curved Shared Boundary Constraint (CSBC)

This constraint relates the orientations of the two surfaces on opposite sides of an edge. It is a generalization of the constraint used in polyhedral scene analysis from the early days [Mackworth, 1973] and has been stated previously in [Shafer *et al.*, 1983, Ulupinar and Nevatia, 1990].

Let two surfaces, S_1 and S_2 intersect along a curve, Γ , whose projection is the curve $\Gamma_1(s) = (\Gamma_x(s), \Gamma_y(s))$. Let the orientations of the surfaces S_1 and S_2 along the curve $\Gamma(s)$, in gradient space, be given by $(p_1(s), q_1(s))$ and $(p_2(s), q_2(s))$. Then CSBC states that:

$$\Gamma'_x(s)(p_2(s) - p_1(s)) + \Gamma'_y(s)(q_2(s) - q_1(s)) = 0 \quad (8)$$

A stronger constraint can be obtained if we can assume that the 3-D intersection curve, Γ , is planar. Say, Γ lies in a plane with orientation (p_c, q_c) . With the assumption of planarity the constraint equation becomes:

$$\Gamma'_x(s)(p_c - p_i(s)) + \Gamma'_y(s)(q_c - q_i(s)) = 0, \quad i = 1, 2 \quad (9)$$

3.2 Inner Surface Constraint (ISC)

The inner surface constraint restricts the relative orientations of the neighboring points, within a surface. Consider a curve $C(t) = (x(t), y(t), z(t))$ on a C^2 surface S . For each point $P \in C$ associate a vector $R \in T_P$ such that

$$\frac{dC}{dt} \cdot dN_R = 0 \quad (10)$$

where T_P is the tangent plane of the surface S at the point P and dN_R is the derivative of the normal N of the surface S in the direction R .

Theorem 6 Inner Surface Constraint: Under orthographic projection, if an image curve C_I is the projection of the curve C on the surface S and $R_I = (r_x, r_y)$ is the projection of the vector R satisfying equation 10, then the change of the orientation, (p, q) , of the surface S , along the curve C , in the $p-q$ space is restricted by the image vector R_I , as:

$$d(p, q)_{C'} \cdot R_I = 0 \quad (11)$$

The proof of the theorem is given in appendix A.

To apply this constraint, we need to identify a curve C in the image plane for which the orientation R can be determined. In a previous paper [Ulupinar and Nevatia, 1990] we have shown that for zero Gaussian curvature surfaces any curve on the surface can be the C curve if the direction R is chosen to be the direction of the rulings of the surface. Following theorem shows how we can use parallel symmetric curves for this purpose in a general case.

Theorem 7 Let the family of curves, $\{C_i\}$, be on a surface S such that the curves, C_i , are parallel symmetric in 3-D. If the curves C_i are used as the C curves of equation 10 then, the tangent of the curves obtained by joining the symmetric points of the curves C_i gives the direction R of the ISC. Conversely, if the curves obtained by joining the parallel symmetric points of curves, C_i , are used as C curves of equation 10 then the tangents of the curves C_i gives the direction R .

Proof Consider the parametric representation $S(u, v)$ of the surface S such that the u parameter curves are parallel symmetric to each other (the $\{C_i\}$ family of curves) and v parameter curves join the parallel symmetric points of the u parameter curves.

For the first part of the theorem we have to show that equation 10 holds or with the current parameterization

$$S_u \cdot N_v = 0 \quad (12)$$

is true, where $N = \frac{S_u \times S_v}{|S_u \times S_v|}$ is the unit normal of the surface. Note that $N \cdot S_u = N \cdot S_v = 0$ by definition. We can substitute $-S_{uv} \cdot N$ for $S_u \cdot N_v$ since:

$$0 = \frac{\partial(S_u \cdot N)}{\partial v} = S_{uv} \cdot N + S_u \cdot N_v \Rightarrow S_u \cdot N_v = -S_{uv} \cdot N \quad (13)$$

S_u is the tangent of the u parameter curves, and since the v parameter curves join the parallel symmetric points of u parameter curves the direction of $S_u(u, v)$ is independent of the v parameter, that is $S_u(u, v) = c(v)S_u(u)$ where c is a scalar function. And;

$$S_{uv} = \frac{\partial S_u}{\partial v} = c'(v)S_u(u) \quad (14)$$

By substituting this in equation 13 we get

$$N_v \cdot S_u = -N \cdot S_{uv} = -c'(v)(N \cdot S_u(u)) = 0 \quad (15)$$

For the second part of the theorem we have to show that $S_v \cdot N_u = 0$. Using equation 15 we get:

$$0 = N_v \cdot S_u = -N \cdot S_{uv} = -N \cdot S_{vu} = S_v \cdot N_u \quad (16)$$

□

3.3 Orthogonality Constraint (OC)

The two previous constraints (CSBC and ISC) are not sufficient to determine surface orientations uniquely. To further constrain the solution, we impose an additional constraint. We require that the cross sections and the meridians of a surface (as defined in sections 2.3 and 2.4) be mutually orthogonal. This constraint may be satisfied precisely for some kinds of surfaces but is not necessarily true for all surfaces; in the latter cases we maximize a measure of orthogonality (given later). This constraint is justified on perceptual observations. It may be viewed as being equivalent to slicing the surface along meridians and cross sections to obtain thin skew symmetric planar patches and assuming that these patches are orthogonally symmetric in 3-D, as in Kanade's analysis for polyhedra [Kanade, 1981]. The orthogonality of two vectors A and B , which lie on a plane having gradient (p, q) and whose images are $A_i = (a_x, a_y)$ and $B_i = (b_x, b_y)$, constrain the gradient (p, q) with the equation:

$$(a_x, a_y, pa_x + qa_y) \cdot (b_x, b_y, pb_x + qb_y) = 0 \quad (17)$$

4 Analysis of ZGC Surfaces

We have applied the constraints of section 3 to analysis of zero-Gaussian curvature surfaces in previous work [Ulupinar and Nevatia, 1990]. We provide a brief summary of this work as the techniques for the PRCGs and SHGCs are related to it.

A ZGC surface is indicated by the presence of a parallel symmetry and a mirror symmetry where lines of symmetry are straight. The parallel symmetry curves give the cross sections of the ZGC and the lines of symmetry give the rulings. For a ZGC, it is necessary only to consider one cross section at a time, as the surface orientations can simply be propagated along the ruling.

Suppose we wish to estimate the surface orientation at n points along the cross section (assumed to be planar), we have $2n + 2$ unknowns ($2n$ for n points, 2 unknowns for the orientation of the cross section itself). ISC and CSBC together provide $2n - 1$ constraint equations, leaving three degrees of freedom undetermined. Introducing the orthogonality constraint (in this case requiring the cross section and rulings to be orthogonal) gives an additional n equations; we now have more equations than unknowns.

These equations are, however, not always independent. We find that for a cylindrical surface, all equations can be satisfied exactly and still one degree of freedom remains for the orientation (p_c, q_c) of the cross section plane (it is constrained to be on a line parallel to the axis of the cylinder in the $p - q$ plane). For more general objects, all equations can not be solved exactly. We choose to satisfy CSBC and ISC exactly and minimize a measure of orthogonality. Unfortunately, this minimization procedure also does not, in general, give a unique answer. The minimum is typically achieved when (p_c, q_c) is along a line in the gradient space and the variations are too small along this line to pick a specific value.

This last degree of freedom is removed by using the 3-D shape of the cross section itself. We make the assumption that the 3-D cross section should be as compact as possible, subject to the limits given by other constraints. Our method to accomplish this consists of fitting an ellipse to the cross section and choosing that orientation that gives the least eccentric ellipse in the back projection subject to the orientation satisfying other constraints (namely, its being on a specific line). Also, we apply a correction to this estimate depending on the quality of ellipse fit to bias the answer away from highly slanted orientations. This algorithm is fully described in [Ulupinar and Nevatia, 1990], an outline is given below.

As (p_c, q_c) is constrained to be on a line, the problem is equivalent to estimating only one parameter, say q_c (without loss of generality, as we can rotate the coordinate system as necessary). Steps in estimating q_c are:

1. First Estimation of q_c : An ellipse is fit to the cross section contour, then the orientation of the circle (p_e, q_e) , that would project as the fitted ellipse is projected on the q axis, on the $p - q$ plane to obtain the first approximation of q_c , call it q_e .

It may be necessary to segment the cross section if, it is complex and repetitive. To achieve this, the concavities of the contour are found and matched. If they match in such a way that the cross section is segmented into similar pieces, then a different ellipse is fit to each piece of the contour and average of the ellipses is used to estimate q_e .

2. Updating q_c : The purpose of this updating process

is to simulate the bias that humans have in orienting the cross section toward 45° . We update q_e to obtain the final q_c as follows (after converting q_e into degrees):

$$q_c = 45^\circ + \lambda(q_e - 45^\circ) \quad (18)$$

Where λ is a confidence factor in the range $[0, 1]$ and is a function of how well the ellipse approximates the cross section curve. In our implementation it is given by :

$$\lambda(\epsilon) = (1 - \epsilon^2) \quad (19)$$

Where ϵ is the ellipse fit error (in range $[0, 1]$).

The algorithm derives from our observations of human perception and we have validated it by an extensive comparison with human subjects.

The described method for recover ZGC surfaces from image contours has been tested on a number of examples (we assume that symmetries are given) and produces results that appear consistent with human observation.

5 Quantitative Shape Recovery of SHGC surfaces

To compute the shape of an SHGC along each recovered cross section curve we can apply the constraints discussed in section 3 as they are applied to a ZGC surface in section 4. For the following; say that there are m cross section curves and we would like to compute the orientation of the surface n points along a cross section. Then we have $2nm$ unknowns, initially, corresponding to the gradient (p, q) of the surface at nm points.

CSBC The curved shared boundary constraint applies between the orientation, (p_c, q_c) , of the cross section curves C_j and the orientation, (p_i, q_i) of each of the point on the surface along a cross section. Note that (p_c, q_c) is the same for all cross section curves. The curved shared boundary states that the line in the $p - q$ space from the gradient (p_i, q_i) of a point $P_i \in C_j$ to the gradient (p_c, q_c) of the cross section plane is orthogonal to the tangent, $C'_j(P_i)$, of the cross section C_j at point P_i . Then the constraint equation is:

$$(p_c - p_i, q_c - q_i) \cdot C'_j(P_i) = 0 \quad \forall P_i \in C_j \quad (20)$$

This provides n constraints along each cross section curve.

ISC Inner surface constraint is applied along a cross section using the tangents of the meridians at each point. The theorem 7 indicates that ISC is applicable along the cross section curves because cross section curves are parallel symmetric by theorem 4 with the meridian curves joining the parallel points of the cross section curves. Inner surface constraint states that change of the orientation $(p_{i+1} - p_i, q_{i+1} - q_i)$ of the surface along a cross section curve C_j between two consecutive points $P_i, P_{i+1} \in C_j$ must be orthogonal to the tangent $M'_{i+1/2}(P_{i+1/2})$ of the meridian that passes through the

point $P_{i+1/2} \in C_j$ which is in the middle middle of the points P_i and P_{i+1} . Then the constraint equation is:

$$(p_{i+1} - p_i, q_{i+1} - q_i) \cdot M'_{i+1/2}(P_{i+1/2}) \quad \forall P_i, P_{i+1/2}, P_{i+1} \in C_j \quad (21)$$

Application of ISC provides $n-1$ equations for each cross section curve.

There are $2n$ unknowns for each cross section curve, and two more unknowns for the whole SHGC, the (p_c, q_c) , by combining the two constraints, we have $2n-1$ constraints for each cross section. Then for each cross section there are three degrees of freedom as in the case of a ZGC surface discussed in section 4.

Planarity of Meridians The meridians of an SHGC are planar as discussed in section 2.3. Then the shared boundary constraint can be applied along a meridian curve as if the curve is obtained by cutting the surface of the SHGC with a plane along the meridian. The shared boundary constraint is applied along a meridian, M , between the gradient, (p_m, q_m) , of the plane that the meridian M rests on and the gradient (p_j, q_j) of the points $P_j \in M$, using the tangent, $M'(P_j)$ of the meridian curve at each point $P_j \in M$. The constraint equation is:

$$(p_m - p_j, q_m - q_j) \cdot M'(P_j) = 0 \quad \forall P_j \in M \quad (22)$$

Enforcing one meridian curve to be planar automatically makes the others to be planar too. Therefore, the planarity is applied only to one of the meridians, giving m constraint equations with the expense of two additional unknowns.

In total there are now $2nm+4$ unknowns, $2nm$ for the (p, q) of nm points on the surface, two for (p_c, q_c) , two more for (p_m, q_m) , and there are $2nm$ constraint equations, nm from the CSBC between the cross sections and the face of the surface, $m(n-1)$ from the ISC, and m from the CSBC of a meridian curve. That is there are four degrees of freedom for recovering the orientation of all the points on an SHGC. These four degrees of freedom corresponds to the orientation, (p_c, q_c) , of the cross sections and the orientation, (p_m, q_m) , of the plane containing the chosen meridian. Without any assumptions we could arbitrarily set these four variables and get a valid reconstruction of the SHGC that would project like the figure in the image plane. However not all of these reconstruction look *natural* to humans when they observe the image of the contours of an SHGC. Humans prefer some interpretations over the others. In the following section we propose orthogonality as the preference criteria.

5.0.1 Orthogonality

For SHGCs we use the orthogonality of the 3-D tangents of the cross sections and the meridian curves, making each little patch, formed by dividing the surface along meridians and the cross sections, orthogonal. We can apply the orthogonality constraint using the equation given in equation 17. This constraint is not always exactly satisfied, except for surfaces of revolution. Therefore we perform a minimization of the second orthogonality constraint as:

$$\Xi = \sum_i \sum_j \cos(\theta_{ij}) = \frac{(C_i(P_{ij}))'_3 \cdot (M_j(P_{ij}))'_3}{|(C_i(P_{ij}))'_3| |(M_j(P_{ij}))'_3|} \quad (23)$$

where $(C_i(P_{ij}))'_3$ and $(M_j(P_{ij}))'_3$ are the 3-D tangents of the cross section and meridian curves at point P_{ij} . These 3-D tangents are dependent on their 2-D tangents on the image and on the orientation (p_{ij}, q_{ij}) of the surface at point P_{ij} as given by the equation 17. The gradients (p_{ij}, q_{ij}) at each point is dependent on the four variables, (p_c, q_c) and (p_m, q_m) , discussed in the previous section. We would like to minimize the function Ξ for (p_c, q_c) and (p_m, q_m) . However from our experiments we observe that minimization of Ξ chooses values that are always consistent with the assumption that the 3-D axis of the SHGC is orthogonal to its cross section.

If we enforce the cross sections to be orthogonal to the axis of the SHGC, the orientation (p_c, q_c) of the cross section lies along a line in the $p-q$ space that passes through the origin and is in the direction of the image of the axis of the SHGC. This constraint also, in effect, enforces the gradient (p_m, q_m) of the plane of the meridians to be orthogonal to the gradient (p_c, q_c) of the cross sections. That is:

$$(p_m, q_m, 1) \cdot (p_c, q_c, 1) = 0 \quad (24)$$

For simplicity, say the coordinate system is rotated such that the image of the axis of the SHGC is aligned with the y axis of the coordinate system. Then, we have; $p_c = 0$ from the orthogonality of the axis to the cross section and $q_m = -1/q_c$ from equation 24. The parameters p_m and q_c are the free variables to be fixed by minimizing the function Ξ . However, the minimum of the function Ξ does not fix the variable q_c (except for surfaces of revolution). Either the function forms a valley along q_c making any choice as good as any other or fixes q_c to be zero which is not a realistic solution. We use the same method for estimating q_c as described for ZGC surfaces in section 4.

5.1 Results

We have implemented the constraints discussed in the previous section in a somewhat reverse order. For an SHGC whose axis is aligned with the y axis of the coordinate system the method is as follows; First the ellipse fit algorithm is applied to compute q_c , then the function Ξ is minimized to compute p_m . Then the surface is constructed using the constraints discussed in section 5 to compute the surface orientation at each point. Figure 11 shows the needle images and the shaded images, with the computed surface orientations, of the SHGCs in figure 1.

6 Quantitative Shape Recovery of PRCGC Surfaces

Here we discuss the application of the three constraints discussed in section 3 along a cross section curve of a PRCGC, to recover the surface orientation of a PRCGC.

CSBC The shared boundary constraint can be applied along the image of a cross section curve. Let (p_c, q_c) be the gradient of the plane that contains the cross section curve, $C(u)$, whose image is the image curve $C_i(u) = (c_x(u), c_y(u))$. Let $(p(u), q(u))$ be the orientation of the points along the cross section curve $C(u)$. Then the

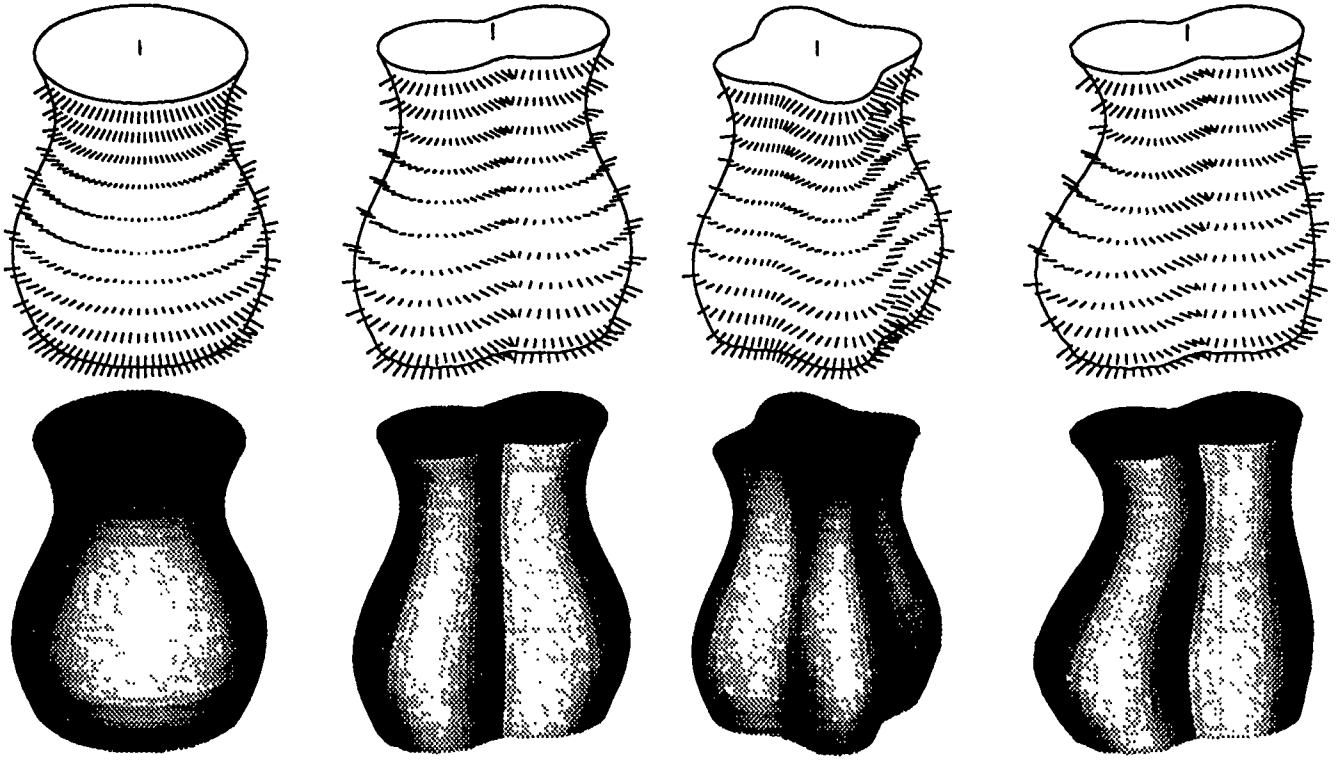


Figure 11: The needle images and the shaded images generated with the computed gradients at each point of the SHGCs in figure 1

shared boundary constraint is:

$$(p_c - p(u), q_c - q(u)) \cdot (c'_x(u), c'_y(u)) = 0 \quad (25)$$

ISC Theorem 7 indicates that ISC is applicable along the cross sections of a PRCGC because cross sections of a PRCGC join the parallel symmetric points of the meridian curves which are parallel symmetric as given by lemma 1. Since the meridians of a PRCGC are parallel symmetric with cross section curves forming the correspondence, the tangent vectors of the meridians along a cross section is a constant vector which is also parallel to the axis of the PRCGC as given by equation 7. Let the tangent direction of the meridians along the cross section $C(u)$ be A' and its image be $A'_i = (a'_x, a'_y)$, note that A'_i is independent of the u parameter. For the sake of simplicity let us assume that the coordinate system is rotated such that A'_i is along the y axis of the coordinate system, then $a'_x = 0$. The inner surface constraint is:

$$\frac{d}{du}(p(u), q(u)) \cdot (a'_x, a'_y) = 0 \Rightarrow q(u)'a'_y = 0 \Rightarrow q(u) = q_0 \quad (26)$$

By combining this constraint with the CSBC given in equation 25 we get:

$$p(u) = \frac{c'_y(u)(q_c - q_0)}{c'_x(u)} + p_c \quad (27)$$

Orthogonality The last constraint is the orthogonality of the meridians to the cross section curves. The reader can easily verify that the u and t parameter curves in equation 3 are orthogonal to each other for all points

on the surface S of the PRCGC. Then, we use the orthogonality by enforcing the tangent of the meridians A' , whose image is $A'_i = (0, a'_y)$ to be orthogonal to the tangent of the cross section curve C , whose image is $C_i(u) = (c_x(u), c_y(u))$, at a point on the surface whose gradient is $(p(u), q_0)$:

$$(0, a'_y, q_0 a'_y) \cdot (c'_x(u), c'_y(u), p(u)c'_x(u) + q_0 c'_y(u)) = 0 \quad (28)$$

By substituting $p(u)$ given in equation 27 in the above equation we get:

$$a'_y c'_y(u)(1 + q_0 q_c) + a'_y p_c c'_x(u) = 0 \quad (29)$$

Since the above equation is zero for all values of u we get both $p_c = 0$ and

$$1 + q_0 q_c = 0 \Rightarrow q_0 = -1/q_c \quad (30)$$

Fixing q_c fixes the orientation of the surface along the cross section C together with the gradient (p_c, q_c) (which is $(0, q_c)$ in the rotated coordinate system) of the plane containing C . However our constraint equations do not constrain q_c .

6.1 Recovering Cross Section Curves

In the previous section we have discussed how to recover the surface orientation at each point on a cross section curve given the image of the cross section curve. However it is not directly possible to replicate the images of the cross section curves of a PRCGC, such as the ones in figure 2, except for the ends of the PRCGC, where we assume the cross section curve is given. That is we assume that the surface is cut along its cross sections.

Here we discuss a method for recovering the cross sections when one or both ends of the PRCGC are available, the method also enables us to reconstruct the 3-D PRCGC from the image of it.

At one end of the PRCGC let the image of the end cross section curve be $C_i(u) = (c_x(u), c_y(u))$ and the image of the axis be $A_i(t) = (a_x(t), a_y(t))$ as in the previous section. The image of the axis at the point it intersect the cross section C is $A_i(0) = (a_x(0), a_y(0))$. Say the coordinate system is rotated such that $a'_x(0) = 0$, and the orientation (p_c, q_c) of the plane containing the cross section curve C is computed using the constraints discussed in section 6 with $p_c = 0$. The orientation of the points along the cross section curve C is (p_u, q_u) where $q_u = -1/q_c$ and $p(u)$ is given by equation 27. Since the meridian curves are parallel symmetric to the axis of the PRCGC we can use the gradient $(p(u), q_u)$ to recover the tangent of the 3-D axis at $t = 0$ as:

$$\begin{aligned} A'(0) &= (a'_x(0), a'_y(0), p(u)a'_x(0) + q_u a'_y(0)) = (31) \\ &= (0, a'_y(0), -\frac{a'_y(0)}{q_c}) \equiv (0, q_c, 1) \end{aligned}$$

That is $A'(0)$ is parallel to normal, $(0, q_c, 1)$, of the plane containing the cross section C , or the plane Π_a containing $A'(0)$ is orthogonal to the plane of C . Also since the axis, A , of the PRCGC is planar the plane Π_a contains the whole axis curve A .

In the following we give an algorithm for recovering the 3-D cross sections from the image of a PRCGC given the gradient (p_a, q_a) of the plane Π_a containing the axis. Then in the next subsection we give a method for computing (p_a, q_a) from the image.

The gradient (p_c, q_c) of the plane of the cross section C can be computed if the gradient (p_a, q_a) of Π_a is given. The gradient (p_c, q_c) must lie on a line that passes through the origin and in the direction of $A_i(0)$, in our case $p_c = 0$, and $(p_c, q_c, 1)$ is orthogonal to $(p_a, q_a, 1)$ then:

$$(0, q_c, 1) \cdot (p_a, q_a, 1) = 0 \Rightarrow q_c = -\frac{1}{q_a} \quad (32)$$

We can compute the 3-D cross section C from the image C_i of it by backprojecting C_i to a plane having gradient (p_c, q_c) .

If the cross section is rotationally symmetric² the algorithm for recovering cross sections is much simpler. In the following we give an algorithm that applies to general, not necessarily rotationally symmetric case.

It can be shown that the image of the axis of the PRCGC, $A_i(t)$, is not always the same as the axis, $B_i(t)$, of the parallel symmetry of the image of the limb edges, where the axis of the PRCGC is the trace of a single point on the cross section as the cross section is swept. This is shown in figure 12. However the image curves $A_i(t)$ and $B_i(t)$ are always parallel symmetric to each other such that the corresponding points are on the same cross section. By using lemma 1 and theorem 5 we conclude that the images of the limb edges are parallel symmetric to

² A planar cross section is rotationally symmetric iff the lines passing through the center of the cross section intersects both sides of the cross section at equal distances.

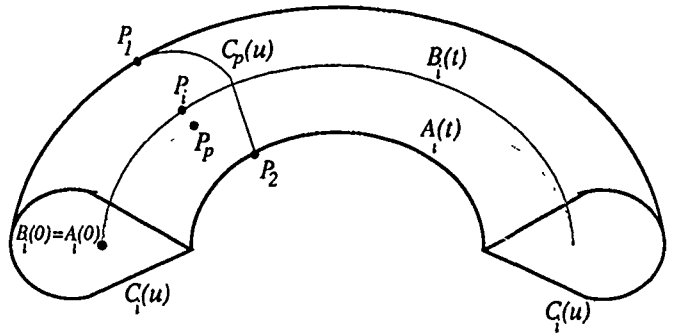


Figure 12: A PRCGC with a non-rotationally symmetric cross section.

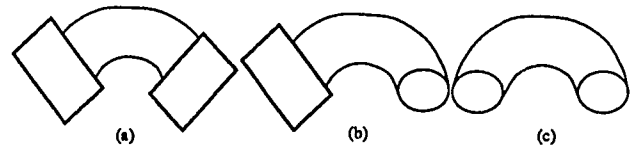


Figure 13: A PRCGC with, (a) none, (b) one, and (c) both end cross sections available.

each other (and of course to its axis) as well as to the images of the meridians of the surface, and meridians of the surface are parallel symmetric to the axis of the PRCGC by equation 7, so are their images. Therefore the axis of the image of the limb edges, the $B_i(t)$ curve, is parallel symmetric to the image of the axis of the PRCGC, the $A_i(t)$ curve.

If we take the axis A of the PRCGC as the trace of the point that is the backprojection of $B_i(0)$ to the cross section plane C . Then $A_i(0) = B_i(0)$. Given the orientation (p_a, q_a) of the plane Π_a containing the axis A , to recover the 3-D cross section say at point P_i on the image axis B_i ; The backprojected C of C_i is rotated by the rotation matrix $R(B'(0), B'(P))$ to obtain the 3-D cross section curve $C_p(u)$ at point P , where $B'(0)$ and $B'(P)$ are obtained by backprojecting $B'_i(0)$ and $B'_i(p_i)$ onto the plane Π_a . Then the points P_1 and P_2 that produces the limb edge on the cross section $C_p(u)$ is identified by equating the image tangents of $C_p(u)$ to the image tangent of limb boundaries P_1 and P_2 . The position of the cross section C_p in 3-D such that $C_p(P_1)$ and $C_p(P_2)$ project as the points P_1 and P_2 on the image and the point P_p on C_p that corresponds to the point $A(0)$ on C is on the plane Π_a , gives the relative position of the cross section C_p with respect to end cross section C in 3-D.

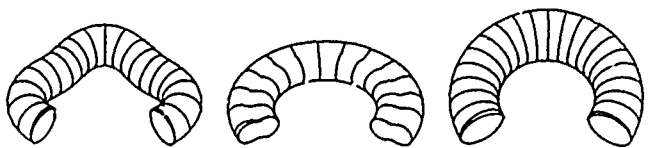


Figure 14: The recovered cross sections for the PRCGCs in figure 2.

6.2 Computing (p_a, q_a)

The gradient (p_a, q_a) of the plane Π_a containing the axis is computed by performing a search in the gradient plane. The objective of the search is to compute (p_a, q_a) that gives a valid reconstruction. A valid construction is one that makes the projection of the cross section points $C_p(P_1)$ and $C_p(P_2)$ exactly the same as the points P_1 and P_2 on the image plane. We form an objective function which is the average distance, on the image plane, of the reconstructed and projected point $C_p(P_2)$ to the point P_2 when $C_p(P_1)$ and P_1 is aligned exactly. Then this objective function is minimized for (p_a, q_a) .

The search is facilitated by finding a good initial point for (p_a, q_a) using the shapes of the end cross sections. The analysis in section 6 show that the gradient (p_c, q_c) of the cross section at one end is constraint to be on a line in the gradient space. A particular value on that line may be chosen by using the ellipse fit discussed in section 4. Similar analysis applies to the other end of the PRCGC (if available). Say the orientation of the plane containing the other end cross section C_n is (p_n, q_n) . Then the plane of C_n is orthogonal to the plane Π_a . If (p_n, q_n) is not equal to $(0, q_c)$ we can compute an initial normal $N_a = (p_a, q_a, 1)$ of Π_a as $N_a \equiv (p_n, q_n, 1) \times (0, q_c, 1)$. If the other end cross section C_n is not available then the gradient (p_a, q_a) is constrained to be on a line by its orthogonality to $(0, q_c)$. The equation of the line containing (p_a, q_a) is $(0, q_c, 1) \cdot (p_a, q_a, 1) = 0$. Any particular value of (p_a, q_a) may be chosen on this line as the initial (p_a, q_a) . Figure 13 shows that perception is more definite when both ends are available, which confirms the above observation that two ends are more informative than one only.

6.3 Results

We have implemented the cross section recovery method described in section 6.1. In the implementation first the orientations (p_c, q_c) and (p_n, q_n) of the end cross sections are computed. Then the normal N_a of Π_a is found by searching around the gradient given by $(p_c, q_c, 1) \times (p_n, q_n, 1)$ that gives a valid reconstruction. The 3-D position of each cross section is then found by translating the end cross section rotating and aligning it with the limb boundaries and the plane of the axis Π_a . Figure 14 shows the recovered cross sections and figure 15 shows the recovered orientations by both needle and shaded images for the PRCGCs given in figure 2.

7 Conclusions

In this paper we have analyzed two class of objects; Straight Homogeneous Generalized Cones (SHGCs) and Planar Right Constant cross section Generalized Cones (PRCGCs).

We show the property of the limb boundaries of SHGCs, under both orthographic and perspective projection, that the tangents of the images of the limb boundaries, if extended from the points on the same cross section, intersect the image of the axis of the SHGC at the same point. We also show that the cross sections

of an SHGC are parallel symmetric, and use that property to recover the images of the cross sections of the SHGC. Then we apply the constraints, curved shared boundary constraint (CSBC), inner surface constraint (ISC), and the orthogonality constraint (OC) to the SHGCs. An SHGC has four degrees of freedom if it is to be recovered from the images of its contours without any assumptions. With the assumption of orthogonality there is only one degree of freedom which is fixed by estimating the orientation of the cross sections with an ellipse fit algorithm. Some computational results are shown on synthetic data.

For PRCGCs the limb boundaries are shown to project as parallel symmetric curves, which enable us to find points on the limb boundaries that correspond to the same cross section. We also show that the three constraints, CSBC, ISC and OC, are applicable along the cross section of a PRCGC. We applied the constraints to the ends where the cross sections are available. Then we present an algorithm to reconstruct the 3-D PRCGC from the images of its contours, using the ellipse fit method to recover the orientations of cross sections at the ends.

We have assumed that the object boundaries and symmetries are given. Detection and computation of symmetries may, in itself, be a difficult task in real images. However, we do provide tests that can be used to verify symmetry properties. Also, we believe that 3-D shape recovery process will serve as an aid in segmentation and boundary labelling process as well. In the future, we hope to explore this aspect of the problem.

Appendix

A Proof of the theorem 6

Let $X(u, v)$ be the local parameterization of the surface S around the point $P \in C(t)$ such that for $P = X(u_0, v_0)$, the curve $X(u, v_0)$ is the curve C and the curve $X(u_0, v)$ is in the direction R . That is, u parameter curve is along the curve C and v parameter curve is in the direction R at the point P . Here we have to show that $\frac{d(p, q)}{du} \cdot R_r = 0$ where (p, q) is the normal of the surface in the gradient space, du is in the direction of C' , R_r is the image, (x_v, y_v) , of the vector $R = X_v = (x_v, y_v, z_v)$ under orthographic projection.

Normal, N , of this surface at any point is given by:

$$N = \frac{X_u \times X_v}{|X_u \times X_v|} \quad (33)$$

Then, the functions dC/dt and dN_R are:

$$\frac{dC}{dt} = \frac{\partial X}{\partial v} = X_v \quad (34)$$

$$dN_R = \frac{\partial N}{\partial u} = N_u \quad (35)$$

By equation 10 we have $X_v \cdot N_u = 0$. Let the normal N of the surface around point P is represented in the (p, q) space as $N = c(p, q, 1)$. Where c is the scale coefficient and equal to $(p^2 + q^2 + 1)^{-1/2}$. Differentiation of N with

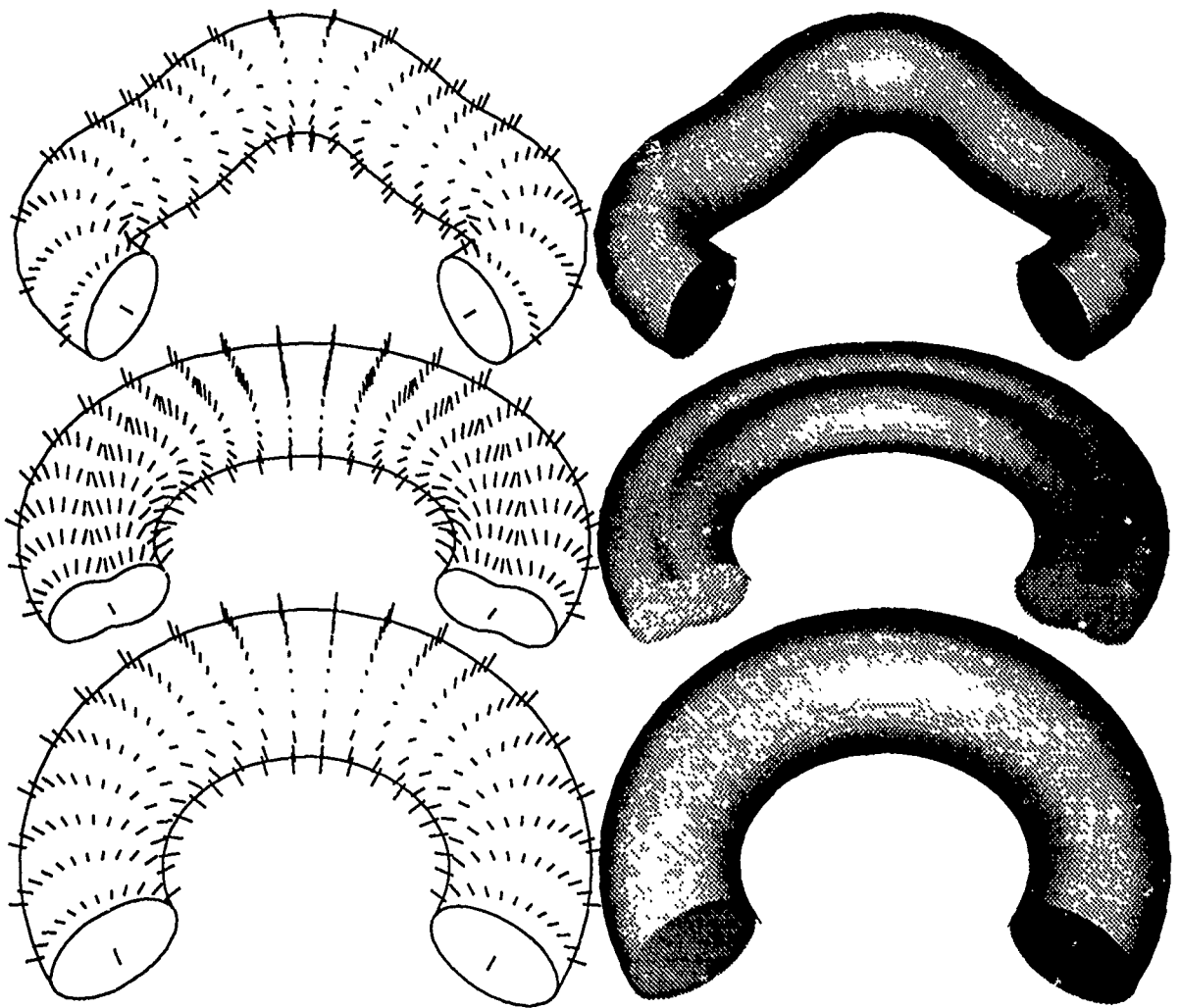


Figure 15: The recovered orientations shown by both needle image and by shading the objects for the PRCGCs in figure 2.

respect to the parameter u gives:

$$\begin{aligned} N_u &= c_u(p, q, 1) + c(p_u, q_u, 0) \\ &= \frac{c_u}{c} N + c(p_u, q_u, 0) \end{aligned} \quad (36)$$

If we set $X_v \cdot N_u = 0$ where $X_v = (x_v, y_v, z_v)$ and N_u is given in equation 36 we get:

$$X_v \cdot N_u = \frac{c_u}{c} X_v \cdot N + c(x_v, y_v, z_v) \cdot (p_u, q_u, 0) = 0 \quad (37)$$

We also have $N \cdot X_v = 0$ from 33. Therefore

$$\begin{aligned} x_v p_u + y_v q_u &= 0 \\ \frac{d(p, q)}{du} \cdot R_1 &= 0 \end{aligned} \quad (38)$$

□

References

- [Barrow and Tenenbaum, 1981] H.G. Barrow and J.M. Tenenbaum. Interpreting line drawings as three dimensional surfaces. *Artificial Intelligence*, 15:105-116, 1981.
- [Clowes, 1971] M.B. Clowes. On seeing things. *Artificial Intelligence*, 2(1):79-116, 1971.
- [Do Carmo, 1976] M. P. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall, 1976.
- [Horaud and Brady, 1988] R. Horaud and M. Brady. On the geometric interpretation of image contours. *Artificial Intelligence*, 37:333-353, 1988.
- [Huffman, 1971] D.A. Huffman. Impossible objects as nonsense sentences. *Machine Intelligence*, 6:295-323, 1971.
- [Kanade, 1981] T. Kanade. Recovery of the three-dimensional shape of an object from a single view. *Artificial Intelligence*, 17:409-460, 1981.
- [Koenderink, 1984] J. J. Koenderink. What does the occluding contour tell us about solid shape. *Perception*, 13:321-330, 1984.
- [Mackworth, 1973] A.K. Mackworth. Interpreting pictures of polyhedral scenes. *Artificial Intelligence*, 4:121-137, 1973.
- [Nalwa, 1987] V. Nalwa. Line drawing interpretation: Bilateral symmetry. In *Proceedings of the Image Un-*

- derstanding Workshop, pages 956-967, 1987. Los Angeles.
- [Nevatia and Binford, 1977] R. Nevatia and T.O. Binford. Description and recognition of complex-curved objects. *Artificial Intelligence*, 8:77-98, 1977.
- [Ponce et al., 1989] J. Ponce, D. Chelberg, and W. B. Mann. Invariant properties of straight homogeneous generalized cylinders and their contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(9):951-966, 1989.
- [Rao, 1988] Kashipati Rao. *Shape Description from Sparse and Imperfect Data*. PhD thesis, University of Southern California, 1988.
- [Shafer and Kanade, 1983] S.A. Shafer and T. Kanade. The theory of straight homogeneous generalized cylinders. Technical Report Report CS-083-105, Carnegie-Mellon University, 1983.
- [Shafer et al., 1983] S. A. Shafer, Kanade T., and J.R. Kender. Gradient space under orthography and perspective. *Computer Vision, Graphics and Image Processing*, 24:182-199, 1983.
- [Shafer, 1983] S.A. Shafer. Shadow geometry and occluding contours of generalized cylinders. Technical Report Report CS-83-131, Carnegie-Mellon University, May 1983.
- [Stevens, 1981] K. A. Stevens. The visual interpretations of surface contours. *Artificial Intelligence*, 17:47-73, 1981.
- [Sugihara, 1986] K Sugihara. *Machine Interpretation of Line Drawings*. MIT Press, 1986.
- [Ulupinar and Nevatia, 1988] F. Ulupinar and R. Nevatia. Using symmetries for analysis of shape from contour. In *Proceedings of the 2nd ICCV*, pages 414-426, 1988. Florida.
- [Ulupinar and Nevatia, 1990] F. Ulupinar and R. Nevatia. Perception of 3-d surfaces from 2-d contours. In *Proceedings of the 10th International Conference on Pattern Recognition*, pages 147-154, 1990. Atlantic City, New Jersey.
- [Xu and Tsuji, 1987] G. Xu and S. Tsuji. Inferring surfaces from boundaries. In *Proceedings of the 1st ICCV*, pages 716-720, 1987. London.

Recovery of Generalized Cylinders from a Single Intensity View

Ari D. Gross and Terrance E. Boulton

Vision/Robotics Lab
Columbia University

New York City, N.Y., 10027 ari@division.columbia.edu, tboulton@cs.columbia.edu

Abstract

Generalized cylinders are a flexible, loosely defined class of parametric shapes capable of modeling many real-world objects. Straight homogeneous generalized cylinders are an important subclass of generalized cylinders whose cross sections are scaled versions of a reference curve. In this paper, a general method is presented for recovering straight homogeneous generalized cylinders from monocular intensity images. This method assumes the generalized cylinder being recovered has purely diffuse reflectance and that the reflectance coefficient is constant. We first demonstrate that contour information alone is insufficient to recover a straight homogeneous generalized cylinder uniquely. The contour image fails to constrain two parameters of the underlying generalized cylinder, the 3D axis slant and translation. A method for recovering the *generalized ruling* of a straight homogeneous generalized cylinder image is then described. Once the rulings of the image have been recovered we show that all parameters derivable from contour alone can be recovered. To recover the two remaining parameters (modulo scale), not constrained by image contour, additional information must be incorporated into the recovery process, e.g. intensity information. We derive a method for recovering the slant of the object using the ruled contour image and intensity values along extremal cross-section curves. In addition, we derive a method for recovering the location of the object's 3D axis from intensity values along meridians of the surface. Using the different methods outlined in this paper constitutes an algorithm for recovering all the shape parameters (modulo scale) of a straight homogeneous generalized cylinder.

1. Introduction

A generalized cylinder (hereafter GC) is a solid defined by its axis, cross-section, and sweeping rule. Generalized cylinders were first proposed by Binford [1] as a class of parametric shapes that is very flexible and capable of modeling many different types of objects. GCs seem general enough to represent many real-world objects yet *sufficiently well-defined* that we are tempted to recover their shape from image data. They have been the topic of considerable research in computer vision and robotics [3],[4],[6],[10],[12],[13],[14]-[18].

Exactly because GCs are such an expressive representation, recovery of their shape parameters from image intensity data has proven to be a difficult problem. As a result, focus has shifted towards important subclasses of generalized cylinders. The subclass that has received the most attention is most likely that of straight homogeneous generalized cylinders (hereafter SHGCs), where the axis is straight and cross-section curves are scaled versions of a reference curve (defined in section 2). But even SHGCs have proven difficult to recover from monocular intensity images. Brooks' ACRONYM system [4] was successful at recovering a very restricted subclass of GCs from contour images. The subclass considered by Brooks in the ACRONYM system consisted of GCs with a circular or simple polygonal cross section, straight or circular spine, and linear or bilinear sweeping rule. Even with this restricted subset of GCs, ACRONYM was only successful at recovering shape from image contour *because it was attempting to match to an a priori set of models*. In fact, a monocular contour image of SHGCs (and certainly of GCs) is insufficient to yield a unique solution, as we show in section 3 of this paper.

There have been other attempts at constructing algorithms for the recovery of SHGCs ([10],[12],[17],[18]-[21]) from a single intensity view. Such attempts generally make use of only contour information. The under-constrained nature of the problem is then compensated for by either considering only restricted classes of SHGCs (e.g., surfaces of revolution), invoking heuristic methods, or having an *a priori* set of models in the database of the recovery system.

In this paper, we provide a method for recovering the *unique* shape of an SHGC (modulo scale) from its intensity image. This is accomplished by first determining the parameters of shape readily available from SHGC contour. The constraints of SHGC contour upon the underlying solid are

presented in this paper in a cursory manner. A more detailed analysis, including relevant theorems and proofs, is presented in [8]. Methods are then developed to recover the remaining parameters, namely, the 3D axis slant and translation, using the SHGC intensity image.

The algorithm presented in this paper relies on some previous results for generalized cylinders [15],[16],[19]. In particular, the algorithm in Ponce et al. [15] for recovering the SHGC image axis has been incorporated into the larger recovery algorithm.

In adding intensity-based methods (sections 5 and 6) to the recovery algorithm, one has to be concerned that the resulting algorithm will require a detailed *a priori* knowledge of the imaging model, such as the number of light sources, their positions and intensities, and the lambertian albedo of the surface. Such a restriction is highly undesirable since, except for highly controlled research environments, such information is generally unavailable. We have tried to avoid this by keeping the assumptions as general as possible. For example, the intensity-based method for slant recovery presented in section 5 makes the following assumptions regarding the imaging model: scaled orthographic projection, lambertian reflectance, and constant albedo. The method, however, does *not* need to know the number of light sources in the imaging model, nor the position and intensity of each light source, nor the lambertian albedo of the material surface. The assumptions are similarly general for the intensity-based method presented in section 6 for recovering the 3D axis position.

This paper avoids some of the assumptions that have been made by generalized cylinder researchers in the past (and have since been shown to be extremely restrictive). In particular, it can be shown that the contour generator of an SHGC is not, in general, planar, nor does it lie along a surface meridian, nor is it symmetric with respect to its axis (see [7],[9],[10],[15]). As a result, we make no such assumptions.

In this paper a *contour generator* is defined as a 3D curve that generates the image contour. There are two kinds of contour generators: limbs, where the surface turns smoothly away from the viewer, and edges, where the surface orientation is discontinuous.

The term *generalized ruling* in the sequel is an extension of the term *ruling* used with respect to ruled surfaces. A *generalized ruling* on an SHGC surface (respectively an SHGC image) are its parallels and meridians (respectively image parallels and image meridians). An example of an SHGC intensity image and its ruled contour is shown in figure 1.

Meridians and parallels, which can be determined directly from the image contour (see section 4), provide a natural parameterization of an SHGC surface and seem to convey considerable information about the underlying shape (as in figure 1b). Nevertheless, in section 3 of this paper we show that, without additional assumptions, no algorithm can recover the shape of an SHGC from the contour image alone. The underlying ambiguity is shown to have two free parameters, slant and location of SHGC 3D axis. The ambiguity is significant and can affect the sign and magnitude of Gaussian curvature at a point on the SHGC surface [8]. Note that this does not imply Gaussian curvature ambiguity along the contour generator itself (see [8],[11]).

In section 2 of this paper, we define generalized cylinders and straight homogeneous generalized cylinders. Next, we show that there exist classes of SHGCs each member of which can generate the identical contour (section 3). In section 4, a method is given for *ruling* the image of an SHGC and it is shown that from the image of a ruled SHGC all parameters of the underlying surface constrained by the contour image can be computed (e.g., the sweeping rule at an image point). In section 5, a method is given for disambiguating among members of a slant contour-equivalent class of SHGCs. This method uses the *intensity* information along an extremal cross-section curve to recover the slant angle of the SHGC object (with respect to the viewer reference frame). Next, a method is presented for recovering the 3D axis position using intensity values at 2 pairs of meridian points (section 6). Finally, the recovery algorithm is implemented on a set of synthetic images (section 7).

2. Generalized Cylinders: Definitions and Assumptions

First, a definition of generalized cylinders as it will be used in this paper.

Definition: A generalized cylinder is the solid swept by a planar cross-section as it is moved and deformed along an axis.

2.1 Straight Homogeneous Generalized Cylinders

Straight homogeneous generalized cylinders have been defined with varying degrees of generality in the literature. Typically, the more general the definition used for SHGCs, the less that can be said about its projective invariant properties and consequently, the harder it is to develop a recovery algorithm. We now define SHGCs as they will be used in this paper.

Definition: An SHGC is a GC with the following properties: the axis is a straight line; the cross-sections are orthogonal to the axis; the cross-sections are deformed only by scaling; the scaling factor can be parameterized as a function of distance along the axis;

Formally, SHGCs can be defined with respect to an orthonormal coordinate system $(O, \vec{i}, \vec{j}, \vec{k})$, where O is a point on the axis, and (\vec{i}, \vec{j}) a vector basis of the reference cross-section plane. We define an SHGC by

$$OP(z, t) = f(z)p(t)\vec{i} + f(z)q(t)\vec{j} + z\vec{k} \quad (2.1)$$

where the function f is the sweeping rule of the SHGC and p and q are the parameterization of the cross-section curve in the \vec{i} and \vec{j} directions, respectively. Note that this definition does not require the SHGC axis to be contained within the closed cross-section curve. SHGCs with the axis external to the cross-section curve do not always appear to have a straight axis. Hereafter, function variables are generally omitted.

Curves on the SHGC surface of constant t are called *meridians* while curves of constant z drawn on the SHGC surface are called *parallels*. This terminology is an extension of that used for surfaces of revolution. It is assumed that both the sweeping rule function and the cross-section curve are twice continuously differentiable (C^2). In the sequel, the terms *image cross-section* and *image meridian* refer, respectively, to the projection of the specified 3D curve onto the image plane.

2.2 The Coordinate System

Consider an SHGC originally aligned with the viewer reference frame, where the viewer reference frame is given by the orthonormal coordinate system $(O, \vec{u}, \vec{v}, \vec{w})$, where \vec{v} is the viewer direction. The SHGC is parameterized in its own coordinate system, having an object-centered orthonormal basis $(\vec{i}, \vec{j}, \vec{k})$. The SHGC is originally in canonical position with respect to the viewer reference frame, i.e., the vectors \vec{u} and \vec{k} , \vec{v} and \vec{i} , and \vec{w} and \vec{j} are respectively parallel (see figure 2.a). Assume the SHGC OP is then rotated in space; the rotated SHGC can be parameterized by

$$OP' = R_\phi R_\psi R_\omega OP \quad (2.2)$$

where ϕ , ψ , and ω are the Euler angles expressing the rotation about the \vec{v} , \vec{w} , and \vec{u} axes respectively, and R_ϕ , R_ψ , and R_ω are the corresponding rotation matrices. Clearly, after the initial rotation around the \vec{u} axis, the resulting SHGC can still be considered in canonical alignment with respect to the $(\vec{u}, \vec{v}, \vec{w})$ viewer-centered reference frame if the initial cross section functions p and q are replaced with new cross-section functions p' and q' (see figure 2.b).

The last rotation, an angle of ϕ around the \vec{v} axis, rotates the projected contour in the image plane but does not modify it in any other way (see figure 2.c). But this rotation in the image plane can be reversed by finding the image axis, as described in [15], where the image axis is a projection of the object-centered \vec{k} axis into the image plane. This rotation of R_ϕ in the image plane can then be *undone* by bringing the image axis into alignment with (parallel to) the viewer-centered \vec{u} axis. Thus, without loss of generality, the only SHGC rotation (from canonical alignment) considered in the sequel is towards or away from the viewer, i.e., around the \vec{w} axis (see figure 2.d). This rotation around the \vec{w} axis is referred to as the SHGC *slant*.

Suppose the viewing direction \vec{v} is given by its spherical coordinates (α, β) in $(O, \vec{i}, \vec{j}, \vec{k})$. Based on the discussion above, without loss of generality, α can be set equal to zero. The resulting orthonormal basis of the viewer reference frame $(\vec{u}, \vec{v}, \vec{w})$ is defined by

$$\vec{u} = -\cos \beta \vec{i} + \sin \beta \vec{k}, \quad \vec{v} = \sin \beta \vec{i} + \cos \beta \vec{k}, \quad \vec{w} = \vec{j} \quad (2.3)$$

Consider the image of an SHGC for some viewing angle, $0 < \beta < \pi/2$. Let $Q(z, t)$ be the projection of surface point $P(z, t)$ onto the image plane. It can be written as

$$OQ(z, t) = f q \vec{w} + (z \sin \beta - f \cos \beta p) \vec{u} \quad (2.4)$$

The projection of a given cross-section curve $z = z_i$ is given by $OQ(z_i, t)$. It is easily seen that each projected cross-section curve is a scaled version of a projected reference curve. This will prove useful for ruling the contour of the SHGC (section 4).

3. Contour-Equivalent Classes

With respect to constraints on the SHGC shape imposed by contour, we want to show that contour-equivalent classes exist among SHGCs, i.e., each member of a class is capable of generating the same contour. Showing that contour-equivalent classes of SHGCs exist is equivalent to showing that there are free parameters of an SHGC unconstrained by contour. In particular, it will be shown that the parameters of axis *slant* and *translation* are unconstrained by contour. For the sake of conciseness, actual proofs of contour equivalence are omitted in this paper. A thorough study of SHGC contour constraints, including proofs of contour equivalence, can be found in [8].

In the next two sections, we formally define two contour-equivalent SHGC classes.

3.1 Slant contour-equivalent SHGCs

The definition for SHGCs is, as in equation (2.1), given by

$$OP(z, t) = f(z)p(t)\vec{i} + f(z)q(t)\vec{j} + z\vec{k} \quad (3.1)$$

Consider a particular SHGC S_0 , defined as above, with sweeping rule $f_0(z)$ and cross-sectional \vec{i} and \vec{j} components of $p_0(t)$ and $q_0(t)$ respectively. We are interested in the contour, both image limbs and edges, generated by SHGC S_0 with slant angle $= \beta$ (as in equation (2.4)). We refer to this contour as $C_{S_0, \beta}$.

Let us define a class of SHGCs, each member of which is capable of generating a contour identical to $C_{S_0, \beta}$. The intuitive idea here is to take the cross-section curve for S_0 , where it is aligned canonically with the viewer's line of sight (as in figures 2.a and 2.b), and stretch it along the viewing direction \vec{v} . We then slant it towards (or away) from the viewer until a contour identical to $C_{S_0, \beta}$ is generated. This, then, motivates our definition for the family of SHGCs that are *slant contour-equivalent* to S_0 , where S_i is *slant contour-equivalent* to S_0 iff $C_{S_i, \beta} = C_{S_0, \beta}$. We define S_i as

$$OP_i(s(z), t) = r(s)k, p_0(t)\vec{i} + r(s)q_0(t)\vec{j} + s(z)\vec{k} \quad (3.2)$$

where $k = \frac{\cos \beta}{\cos \gamma}$, $s(z) = z \frac{\sin \beta}{\sin \gamma}$, $k_i \geq 1$, and $r(s(z)) = f_0(z)$. The k term in equation (3.2) can be viewed as the stretching factor, i.e., the factor by which the cross-section curve for S_i has been stretched in the \vec{i} direction. It is easy to see that, given basis SHGC S_0 and stretching factor k , SHGC S_i is well-defined.

It can be proven [8] that, for any stretching factor $k \geq 1$, the viewing angle γ , and the SHGC S_i , defined in equation (3.2), produce an image contour $C_{S_i, \gamma}$ such that $C_{S_i, \gamma} = C_{S_0, \beta}$, i.e., the SHGCs S_0 and S_i are *slant contour-equivalent*. That being true, one cannot ascertain more about the underlying SHGC, given a monocular image contour (and without using heuristics), than that it is a member of the contour-equivalent class defined above.

Consider the 3 SHGCs shown in figure 3. It can be seen that they have the same contour. Yet, their intensity images are different. When these three SHGCs are seen from the side, as shown in figure 4, it is clear that they are quite different in shape. Next, consider the pair of SHGCs shown in figure 5. In this figure, the two SHGCs have identical contours but differ in their intensity images. A side view of these two SHGCs is given in figure 6, where it can be seen that they differ in their slant towards the viewer and in the eccentricity of the cross-section curve.

3.2 Axis-translated contour-equivalent SHGCs

Consider the SHGC S_0 , defined in the previous section. Again, we are concerned with the contour (both limbs and edges) produced by SHGC S_0 , having sweeping rule $f_0(z)$ and cross-sectional \vec{i} and \vec{j} components of $p_0(t)$ and $q_0(t)$ respectively, with slant $= \beta$ and translation of zero between the cross-section curve's origin and the axis origin. We refer to this contour as $C_{S_0, \beta, 0}$.

Let us define a class of SHGCs, each member of which is capable of producing a contour identical to the one produced by S_0 when viewed from the same viewing angle, i.e., slant angle is equal to β , but where the 3D location of the axis is allowed to vary (as long as it lies in the plane determined by the SHGC image axis). The intuitive idea is to take each cross-section curve of basis SHGC S_0 , translate it with respect to the SHGC axis, and then slide it up or down the axis to achieve a contour equivalent to that of $C_{S_0, \beta, 0}$. This motivates a definition for the family of SHGCs that are *axis-translated contour-equivalent* to S_0 , where S_i is considered *axis-translated contour-equivalent* to S_0 iff $C_{S_i, \beta, 0} = C_{S_0, \beta, 0}$. We define S_i , a member of this contour-equivalent class, as

$$O\vec{P}_i(s(z), t) = r(s)(p_0(t) + h_i)\vec{i} + r(s)q_0(t)\vec{j} + s(z)\vec{k} \quad (3.3)$$

where h_i is the distance the reference cross-section curve's origin has been translated from the axis origin O in the positive \vec{i} direction, and $s(z) = z - \frac{h_i(1-f_0)}{\tan \beta}$, where f_0 is the sweeping rule of SHGC S_0 .

It can be proven [8] that, for any axis translation factor h , the viewing direction with slant $= \beta$ and the SHGC S_i defined in equation (3.3), generate an image contour $C_{S_i, \beta, \lambda}$ such that $C_{S_i, \beta, \lambda} = C_{S_0, \beta, \lambda}$, i.e., the SHGCs S_0 and S_i are *axis-translated contour-equivalent*.

Consider the two SHGCs shown in figure 7. They have the same image contour, yet their respective intensity images are different. An oblique view of this SHGC pair is shown in figure 8, where it can be seen that the right SHGC is a banana-like shape while the left SHGC is a vase-like shape.

Thus, without heuristic methods, we are confronted with the fact that SHGC contours belong to contour-equivalent classes. In particular, there are two degrees of freedom, axis slant and translation.

4. Ruling over generalized cylinders

In this section, a method is described for finding image parallels and meridians from the SHGC contour. This method is referred to as recovering the *generalized ruling* of the SHGC contour, or simply as *ruling* the contour image (see section 1).

In general, most non-occluded image cross-section curves tend to have two or more points of intersection with the image limb, which suggests a method for ruling the SHGC contour. This method assumes that the image axis has already been recovered [15]. Algorithms using the SHGC contour to recover the image axis of an SHGC, where the cross-section function is assumed polar with respect to the axis, are given in [15] (though the robustness of such algorithms is not assured). In [7] the 2D axis lemma, on which the algorithm for recovering the image axis is based, is extended to SHGCs with arbitrary, simple C^2 cross-sections, as defined in equation (3.1).

Once the image axis has been recovered, the reference curve can be *scaled* with respect to the axis so that it touches the bounding contour at 2 or more points, without any point extending beyond the contour. Using a non-accidentalness alignment criterion, we assume that if the above scaling exists, it indicates that the image parallel at this point (along the axis) has been correctly scaled. This method allows us to draw image parallels at any desired point along the SHGC contour. Connecting corresponding points of image parallels together using interpolating splines provides an approximation to the image meridians.

This technique is illustrated in figure 9 which shows the contour and image axis of an SHGC. Also shown in the figure are scaled versions of the image cross-section with respect to a certain point along the image axis. It can be seen that only at one such scaling does the image cross-section curve exactly touch the contour in two places; for every other scaling, the cross-section curve is either contained entirely within or extends beyond the bounding image contour. Thus, this *osculating* image parallel is *presumed* to be the correct scaling of the image cross-section at this point along the image axis. This method works when the SHGC axis is contained within the cross-section curve. In a case where the axis is not contained within the cross-section curve, a more general method is required [7] that involves both rescaling and translating the image cross-section.

The possible parameterizations of an SHGC can be grouped into equivalent classes, as explained in [16]. That being the case, one can decide on a particular parameterization from amongst this class by, somewhat arbitrarily, setting the scaling function of the top cross section curve to one. Having done so, it is clear from equation (2.4) that the scaling factor is also known for all image parallels detected by the method described above, since they are all just scaled versions of a projected reference curve.

5. Solving for the slant angle

In the previous section, a method was given for ruling the SHGC contour. The parameters that remain unconstrained by ruling the SHGC image are axis slant and translation. In this section, a method is described for recovering axis slant. The assumptions this intensity-based method relies on (as well as the method described in section 6) are exactly those mentioned in section 1, namely, scaled orthographic projection, lambertian reflectance, and constant albedo. The method, however, does *not* need to know the number of light sources in the imaging model, nor the position and intensity of each light source, nor the lambertian albedo of the material surface. It is further assumed, however, that the SHGC has an extrema of the sweeping rule (i.e., the sweeping rule is non-monotonic), and no self-shadow.

Consider the SHGC image shown in figure 10a. First, the image axis is recovered (see [15] and [7]). Next, we find contour points that are extrema of distance between the SHGC contour and the image of the SHGC axis. Due to a lemma by Ponce [15], we know that this contour distance extrema corresponds to an extrema of the sweeping rule function f . This lemma is shown to generalize to the case of non-polar SHGCs defined in equation 2.1 [7]. It can easily be verified that the image parallels connecting corresponding contour extrema are projections of planar, lateral geodesic curves, where the surface tangent vector in the meridional direction is parallel to the SHGC axis.

Using the method for ruling the image surface described in section 4, the image parallel connecting a pair of image contour extrema can be recovered. It is easy to show that there exists at least one point on this image parallel that has its tangent vector along the image parallel curve perpendicular to the image axis. We refer to one such image point as C_0 and the image tangent vector at this point as t_0 . Next, we find a pair of image points, one on each side of C_0 , such that the tangents along the image parallel at these points intersect the tangent t_0 at angles θ and $\pi - \theta$, respectively. We refer to these points as C_1 and C_2 , see figure 10a. Let S_0, S_1 , and S_2 be points on the surface of the SHGC that project onto image points C_0, C_1 , and C_2 . It can be shown [7] that the tangent planes at S_1 and S_2 make equal angles with the tangent plane at S_0 , as shown in figure 10b.

Consider a single light source at some arbitrary position in space, though distant from the SHGC object (i.e., projection is scaled orthographic). We now select a new object-centered coordinate system for the SHGC, obtained by rotating the $(\vec{i}, \vec{j}, \vec{k})$ coordinate frame around the \vec{k} axis until the \vec{j} axis is aligned with the normal vector at S_0 . In this new orthonormal coordinate system $(\vec{i}', \vec{j}', \vec{k}')$, the unit normals at S_0, S_1 , and S_2 are given by $(0, 1, 0)$, $(\sin \delta, \cos \delta, 0)$, and $(-\sin \delta, \cos \delta, 0)$, respectively, where δ is the angle made by the intersecting tangent planes at S_1 (or S_2) and S_0 .

We are interested in the image intensity values at C_0, C_1 , and C_2 . Let $\vec{L} = e_1\vec{j}' + e_2\vec{j}' + e_3\vec{k}'$ be the point source directional vector. The image intensity values at C_0, C_1 , and C_2 are directly proportional to the cosine of the angle between \vec{L} and the normal vectors at S_0, S_1 , and S_2 , respectively. Let λ_S be a constant equal to the diffuse reflectance coefficient of the surface, and $I_L = \sqrt{e_1^2 + e_2^2 + e_3^2}$ is the intensity of the incident light. Using Lambert's law, the image intensities at C_0, C_1 , and C_2 are given by

$$R_0 = \lambda_S e_2 \quad (5.1)$$

$$R_1 = \lambda_S (e_1 \sin \delta + e_2 \cos \delta) \quad (5.2)$$

$$R_2 = \lambda_S (-e_1 \sin \delta + e_2 \cos \delta) \quad (5.3)$$

where R_i is the image intensity at image point C_i . Adding R_1 and R_2 together and dividing by R_0 , we have

$$\frac{R_1 + R_2}{R_0} = 2 \cos \delta \quad (5.4)$$

where δ is the angle between the tangent planes at S_0 and S_1 (alternatively, the angle between the tangent planes at S_0 and S_2). Thus, the value of δ can be obtained directly from the intensity values at C_0, C_1 , and C_2 . But knowing the angle δ between the tangent planes and the angle θ between the corresponding image tangents allows us to compute axis slant angle β since $\cos \beta = \frac{\tan \theta}{\tan \delta}$.

This method, then, computes axis slant with respect to the viewer reference frame without knowledge of light source position, diffuse reflectance coefficient of the surface, or intensity of the incident light. Also, the method is equally valid for multiple light sources (see [7]), so long as the 3 surface points S_0, S_1 , and S_2 used for the slant computation are visible to the same set of light sources. An example of this slant recovery method is given in section 7.

6. Recovery of the 3D Axis Position

In this section, an intensity-based method is described for recovering the 3D axis translation parameter h (defined in equation (3.3)). The assumptions required for this method include those described in the previous section and, in addition, require that the SHGC have a planar top (or at least a rim) visible in the image.

The SHGC we want to recover can be parameterized by equation (3.3). We assume at this point in the recovery process that the image axis has already been recovered, the SHGC image has been ruled (section 4), and the slant parameter β has been recovered (section 5). We can now, somewhat arbitrarily, select a point on the image axis O_i . It is easy to see that, with respect to O_i , the values of $p(t)$ and $q(t)$ for any point along the image cross-section can be computed (since axis slant and cross-section scaling are known).

Consider a point source directional vector \vec{L} decomposed into orthogonal components, \vec{L}_a in a direction parallel to the SHGC axis \vec{k} and \vec{L}_p parallel to the SHGC cross-section plane, as shown in figure 11.

Next, consider two points on the SHGC surface, S_{G0} and S_{M0} , which correspond, respectively, to a point on a cross-sectional, lateral surface geodesic (where such points can be found directly from image contour, as in the previous section), and another point on the same surface meridian (see figure 12). We know that the tangent plane at S_{G0} is parallel to the SHGC axis [5], and we would like to determine, as a first step in solving for translation parameter h , the angle that the tangent plane at S_{M0} makes with the tangent plane at S_{G0} . To do this, we divide \vec{L}_p into orthogonal components. Let \vec{L}_m be the vector component of \vec{L}_p parallel to the surface normal vector at S_{G0} , as shown in figure 12. Let \vec{L}_i be the vector component of \vec{L}_p perpendicular to \vec{L}_m . Clearly, the \vec{L}_i component of \vec{L} has no effect on the intensity values at either S_{G0} or S_{M0} since \vec{L}_i is parallel to the tangent planes at both points (see figure 12). Without loss of generality (see [7]), assume that the magnitude of the point source directional vector and the diffuse reflectance coefficient are both equal to 1. The intensity at S_{G0} is then given by

$$R_{G0} = L_m \quad (6.1)$$

while the image intensity at S_{M0} is clearly given by

$$R_{M0} = \cos \phi L_a + \sin \phi L_m \quad (6.2)$$

where ϕ is the angle the tangent plane at S_{M0} makes with the tangent plane at S_{G0} , as shown in figure 13.

We now consider a third point S_{T0} , lying on the planar top (or rim) of the SHGC, with intensity value given by

$$R_{T0} = L_a \quad (6.3)$$

As can be seen in figure 13 (see also [7]), the intensity equation at S_{M0} given by equation (6.2) can be rewritten as

$$R_{M0} = \cos \phi R_{T0} + \sin \phi R_{G0} \quad (6.4)$$

where R_{T0} , R_{G0} and R_{M0} are simply the observed image intensities, respectively, at image points C_{T0} , C_{G0} , and C_{M0} corresponding to surface points S_{T0} , S_{G0} , and S_{M0} .

From equation (6.4) and the intensities at the points S_{T0} , S_{G0} , and S_{M0} , a closed form solution can be obtained for ϕ . There are, in fact two solutions for ϕ , given by

$$\phi = -\frac{R_{G0} \sqrt{R_{G0}^2 + R_{T0}^2 - R_{M0}^2} \pm R_{T0} R_{M0}}{R_{T0} \sqrt{R_{G0}^2 + R_{T0}^2 - R_{M0}^2} \pm R_{G0} R_{M0}} \quad (6.5)$$

This double solution is not a problem in general, as will soon be explained, and we are able to arrive at a unique solution for ϕ .

Assuming, without loss of generality (see section 2.2), that the only rotation of the object is towards (or away) from the viewer, we obtain the following equation for the normal of a point on the SHGC surface

$$\vec{N} = q'\vec{i} - p'\vec{j} + f'(p'q - (p+h)q')\vec{k} \quad (6.6)$$

From the equation for the normal, we can immediately obtain an equation for ϕ . This angle ϕ can be computed using the equation

$$\tan \phi = \frac{(f'(p'q - (p+h)q'))}{\sqrt{q'^2 + p'^2}} \quad (6.7)$$

But we can solve for ϕ directly from the intensity image using equation (6.5). In addition, we assume that the values of p , q , p' , and q' are either known or computable. This is a straightforward procedure as the slant of the object has already been recovered (section 6), so that p and q at a point on the image are directly available from the image, while p' and q' can be

computed by fitting points locally with a polynomial then analytically computing the tangent vector. Thus, the only values in equation (6.7) whose values are still not computable are f' and h .

We are interested in recovering h . To do so, we first rewrite equation (6.7) with f' on the left side, such that

$$f' = \frac{\tan \phi \sqrt{q'^2 + p'^2}}{((p'q) - (p+h)q')} \quad (6.8)$$

To solve for h , we need to select another set of three points on the surface of the SHGC; we use the point S_{T0} again, and additionally select a geodesic point S_{G1} (on the same extremal cross-section curve as S_{G0}) and a point S_{M1} on the same meridian as S_{G1} and on the same cross-section curve as S_{M0} . This set of three points, S_{G1} , S_{M1} , and S_{T0} will yield two solutions for ϕ , as given in equation (6.5). Selecting one of the solutions for ϕ , the tangent plane angle, for the surface points at S_{M0} and S_{M1} and using equation (6.8) gives us two equations for f' , where the only unknown parameter on the right side of the equation is h . Since the points S_{M0} and S_{M1} lie on the same cross-section curve, the value of f' at both points is the same; thus, the right sides of the respective equations for f' generated using equation (6.8) must be equal. Setting them equal, we are able to get a solution for h given by

$$h = \frac{\tan \phi_1 \sqrt{q_1'^2 + p_1'^2} (p_0' q_0 - p_0 q_0') - \tan \phi_0 \sqrt{q_0'^2 + p_0'^2} (p_1' q_1 - p_1 q_1')}{\tan \phi_0 p_1' \sqrt{q_0'^2 + p_0'^2} - \tan \phi_1 p_0' \sqrt{q_1'^2 + p_1'^2}} \quad (6.9)$$

where ϕ_0 is a solution for ϕ from the first set of meridian points (S_{G0} and S_{M0}), ϕ_1 is a solution for ϕ from the second set of meridian points (S_{G1} and S_{M1}), and p_i , q_i , p_i' , and q_i' correspond to the cross-section functions (as defined in equation (3.3)) and their derivatives at the point S_{M_i} . Since every term on the right side of this equation can be computed, the value of h can now also be computed.

To get a unique value for h , since there are two values of ϕ computed for each set of meridian points (corresponding to the two solutions for the slant angle given in equation (6.5)), we take several sets of meridian points, where each set gives us four solutions for h , since there are four ways to select one ϕ value from each pair of ϕ values. Only one solution for h will appear in all sets of solutions for h , and that is the desired value for h . It can be shown [7] that this method works regardless of the number of light sources so long as a light source seen at point S_{G_i} is also seen at point S_{M_i} .

Thus, it has been demonstrated that all shape parameters of the SHGC (in Julo scale) can be computed using constraints from both the contour and the intensity of the SHGC image. An example of the recovery method is given in the next section.

7. Experimental Results

Consider the SHGC shown in figure 14a. We want to recover its slant parameter. Since SHGC slant cannot be ascertained from contour alone, as described in section 3, we will try to recover the slant value using intensity information. The ruled contour and image axis for this SHGC are shown in figure 15. For this experiment, the intensity image and its ruled contour image (with image axis) were given as input to the algorithm.

In order to recover the slant, we first need to find a contour extrema with respect to the image axis. Having done this, the algorithm recovers the image parallel that touches this point along the SHGC contour. A point along this extremal image parallel that has a tangent perpendicular to the axis direction is then found. This can be done reliably by locally approximating the parallel curve with a polynomial and then computing the tangent to the curve at the point analytically. This point we label C_0 , corresponding to the point C_0 in section 5 (see also figure 10a). This image point has the property that, when vertically aligned with respect to the viewer reference frame, its corresponding surface point S_0 (shown in figure 10b) has a value of $p = 0$, $q = 0$ in the gradient space (with respect to the viewer reference frame). We now need to find two other image points, C_1 and C_2 , that make equal angles (with opposite sign) with the tangent line at C_0 . These two points correspond, once again, to C_1 and C_2 of section 5 (as in figure 10a). To select these points, this implementation of the method asks the user to specify an angle the tangents at C_1 and C_2 should make with the tangent at C_0 . When 15 degrees was selected, the algorithm found the image meridians shown in figure 16. The intersection of the image meridians with the image parallel extrema give us the desired points C_1 and C_2 . The middle image meridian corresponds to points along the image parallels whose tangents are perpendicular to the axis. The intersection of this image meridian with the parallel extrema curve is at point C_0 .

The slant is computed in stages (section 5). First, the cosine of angle δ is computed by adding the image intensities at C_1 and C_2 and dividing by twice the image intensity of C_0 , as given in equation (5.4). The δ angle corresponds to the angle between the tangent planes at S_0 and S_1 (alternatively, at S_0 and S_2), as shown in figure 10b. In our example of figure 14a, the algorithm computed a value of $\delta = .59244$ radians, corresponding to 33.94 degrees. The angle between the tangent lines at C_0 and C_1 (alternatively, at C_0 and C_2) was already estimated to be 15 degrees. Having obtained values for θ and δ , the algorithm is able to compute axis slant β since $\cos \beta = \frac{\tan \theta}{\tan \delta}$. The value for axis slant is computed as $\beta = 17.03$ degrees. The error (actual slant was 20 degrees) results from several factors, primarily image estimates of tangent direction and aliasing. The recovered SHGC is shown in figure 14. A side view of the original (left) and recovered (right) SHGCs is shown in figure 17. It seems to be a fairly good recovery of the extent (eccentricity) of the SHGC, which is a function of its axis slant.

Consider the SHGC shown in figure 18a. It looks like a vase from its contour image, however, the intensity information makes it appear that one side is of this vase is rather flat. In fact, this is the case as can be seen from a side view of this SHGC, shown in figure 19a. Our intensity based algorithm should be able to solve for this translation between the origin of the cross-section curve and the origin of the axis (where the axis intersects the top cross-section curve). If we let the origin of the 3D cross-section curve be its centroid then its projection can be determined in the image (i.e., for a planar curve under orthography, the projection of the centroid is the centroid of the projected curve). It is assumed that the slant value is known at this point in the recovery process and only the axis translation needs to be solved for. The values for p and q are computed with respect to the cross-section origin. The values for p' and q' can be approximated, as described above. Input to the algorithm was the intensity image for the SHGC shown in figure 18a and its ruled contour image, with the recovered image axis. The algorithm then found a parallel extrema with respect to the image axis. To arrive at a unique solution for h , the user specifies the number of meridional sets of points to be considered. In this case, we chose 4 sets of points. Using the method described in section 6, the algorithm recovered the SHGC shown in figure 18b. A side view of this SHGC, as compared to the original, is shown in figure 19b. The actual translation value was $h = 0.95$ (where at $h = 1$ the SHGC axis would have intersected the cross-section curve). The recovered value was $h = 0.91$.

8. Conclusion and Future Work

In this paper, an algorithm to uniquely recover the shape of an SHGC (modulo scale) was presented. It incorporates methods using both contour and intensity information. The methods that are intensity-based do not require a knowledge of the number of light sources, their positions, nor their intensities.

The problem of recovering the shape of an underlying SHGC from its contour image is inherently underconstrained. This point is formalized in the paper by defining two classes of contour-equivalent SHGCs. In particular, it was shown that two degrees of freedom exist in an SHGC contour image, namely, axis slant and translation. A method was then given for recovering the generalized ruling of an SHGC image.

This paper shows that additional constraints are required in order to recover SHGC shape from contour. One available constraint is image intensity information. The paper describes a method that makes use of both contour and intensity information to recover the slant of an SHGC (section 5). The method detects extremal image parallels. Intensity values at certain points along these parallel extrema are then used to compute the angle between the tangent planes at corresponding surface points. The angle made by the tangent planes is then used to compute the slant of the SHGC with respect to the viewer reference frame. A method for recovering the axis translation was then described (section 6). This method uses the intensity values at several points on the SHGC image (2 sets of meridian points and a point from the top cross-section plane). Thus, an algorithm for recovering the shape of an SHGC from its intensity image was presented. In section 7, experimental results are given for the slant and translation components of the algorithm.

Further research in this area will take several tracks. First, the algorithm presented in this paper, already tested experimentally, needs to be incorporated into a real-world system robust under noisy imaging conditions. A second direction for further research is to remove some of the restrictions on the current method, e.g., lambertian reflectance and an extrema of the sweeping rule. Finally, the algorithm to recover SHGCs presented in this paper should be extended (if possible) to include a broader class of

GCs, such as one that allows for arbitrary sweeping rules along orthogonal directions of the cross-section curve.

Acknowledgements

Support for this work was provided in part by the Advanced Research Projects Agency of the Department of Defense under contract #N00039-84-C-0165 and by the National Science Foundation Grant #IRI8800370. Thanks to Cliff Beshers for the graphics package used in generating some of the images for this paper.

References

- [1] Binford, T. O., Visual perception by computer, *Proceedings IEEE Conf. on Systems and Control*, Miami, December, 1971.
- [2] Brady, J.M., and Yuille, A., An extremum principle for shape from contour, MIT, AI Lab., MIT-AIM 711, 1983.
- [3] Brady, J.M., Ponce, J., Yuille, A., Asada, H., Describing surfaces, in *Proceedings of the 2nd Int'l Symposium on Robotics Research*, Harasuja and Inoue (ed.), MIT Press, 1985.
- [4] Brooks, R. A., Symbolic reasoning among 3-D models and 2-D images, *Artificial Intelligence* 17, 285-348, 1981.
- [5] do Carmo, M.P., *Differential geometry of curves and surfaces*, Prentice-Hall, Inc., 1976.
- [6] Fearing, F.S., Tactile sensing, perception, and shape interpretation, PhD thesis, Dept. of Electrical Engineering, Stanford University, Dec. 1987.
- [7] Gross, A.D., *Recovery of straight homogeneous generalized cylinders from a single intensity view*, CS Technical Report CUCS-494-89, Columbia University, 1989.
- [8] Gross, A.D., *Straight homogeneous generalized cylinders: Constraints from contour*, submitted to the Proceedings of the Darpa Image Understanding Workshop, Pittsburgh, 1990.
- [9] Gross, A.D., Boulton, T.E., Straight homogeneous generalized cylinders: Analysis of reflectance properties and a necessary condition for class membership, *Darpa IU Workshop*, Palo Alto, CA, 1989.
- [10] Horaud, R., and Brady, J.M., On the geometric interpretation of image contours, in *Proc. First Int'l Conference on Computer Vision*, London, U.K., June 1987.
- [11] Koenderink, J.J., What does occluding contour tell us about solid shape?, *Perception*, vol. 13, pages. 321-330, 1984.
- [12] Marr, D., Analysis of occluding contour, *Proc. of Royal Society of London B-197*, pp. 441-475, 1977.
- [13] Marr, D., *Vision*, San Francisco, CA, WH Freeman, 1982.
- [14] Nalwa, Vic, Line-drawing interpretation: Bilateral symmetry, in *Proceedings of the Image Understanding Workshop*, vol. 2, Feb., 1987, pp. 956-967.
- [15] Ponce, J., Chelberg, D., and Mann, W., Invariant properties of straight homogeneous Generalized Cylinders and their Contours, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. II, No. 9, September 1989.
- [16] Ponce, J., Straight homogeneous generalized cylinders: Differential geometry and uniqueness results, in *Proceedings of Computer Vision and Pattern Recognition Conference*, 327-334, June 1988.
- [17] Rao, K., and Medioni, G., Useful geometric properties of the generalized cone, in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, June 5-9, 1988, pp. 276-281.
- [18] Richetin, M. Dhome, M., and LaPrete, J.T., Inverse perspective transform from zero-curvature curve points application to the localization of some generalized cylinders, in *Proceedings of the 1989 Computer Vision and Pattern Recognition Conference*, San Diego, California, June 4-8, 1989.
- [19] Shafer, S., *Shadows and silhouettes in computer vision*, Kluwer Acad Publishers, Boston, 1985.
- [20] Ulupinar, F., and Nevatia, R., Using symmetries for analysis of shape from contour, in *Proceedings of the Second International Computer Vision Conference*, 414-426, Dec. 5-8, 1988.
- [21] Ulupinar, F., and Nevatia, R., Inferring Shape from contour for curved surfaces, in *Proceedings of the 10th Int'l Conference on Pattern Recognition*, 147-154, June 16-21, 1990.

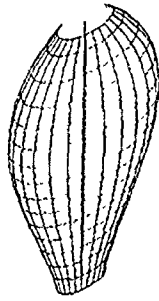


Figure 1. A straight homogeneous generalized cylinder: a. the intensity image; b. the ruled contour.

Coordinate Systems

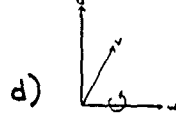
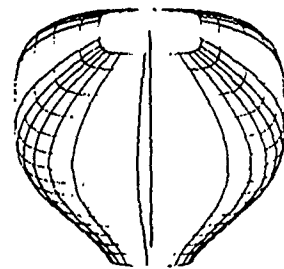
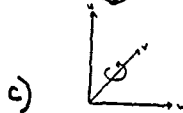
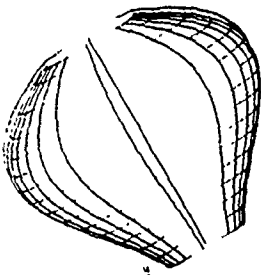
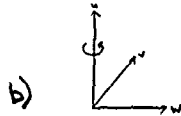
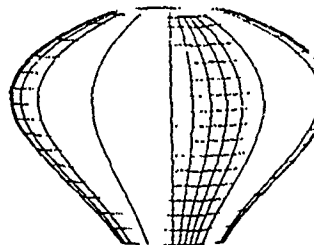
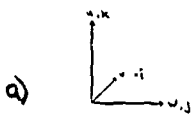
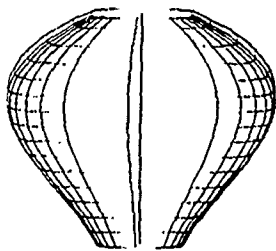


Figure 2. The coordinate system used to define a straight homogeneous generalized cylinder: a. in canonical position ; b. rotated around the u axis ; c. rotated around the v axis ; d. rotated around the w axis, i.e. tilted an angle of β towards the viewer.



Figure 3. A slant contour-equivalent view of 3 vase SHGCs.

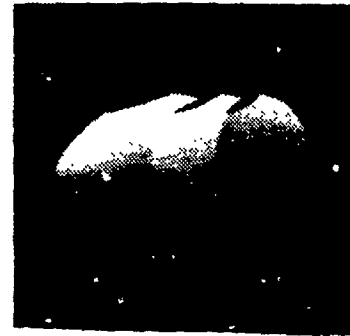


Figure 4. A side view of same 3 vase SHGCs.

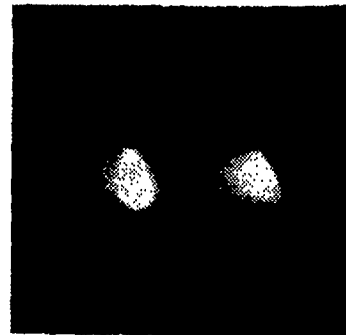


Figure 5. A slant contour-equivalent view of 2 SOR-shaped SHGCs.

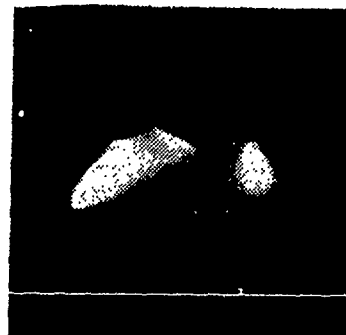


Figure 6. A side view of same 2 SOR-shaped SHGCs.

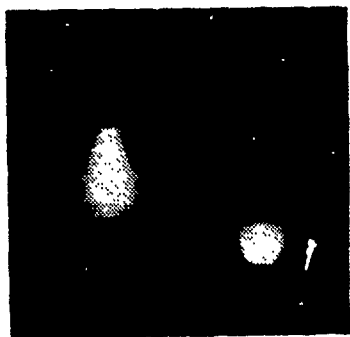


Figure 7. A translation contour-equivalent view of two SHGCs.



Figure 8. An oblique view.

Ruling Over SHGCs

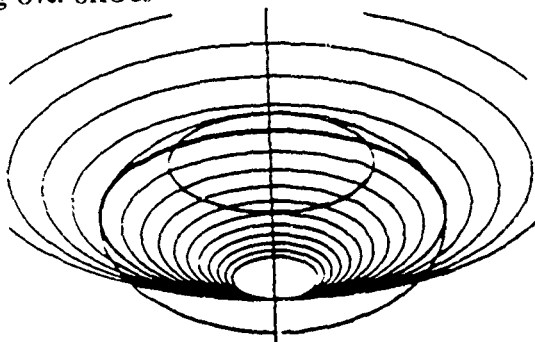


Figure 9. A method for ruling the SHGC image: different scalings of the image parallel and the correct scaling.

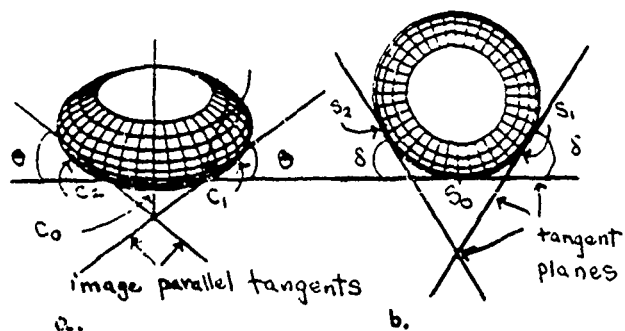


Figure 10. SHGC tilt recovery method: a. image tangents at C_0 , C_1 , and C_2 ; b. overhead view of intersecting tangent planes at surface points S_0 , S_1 , and S_2 .

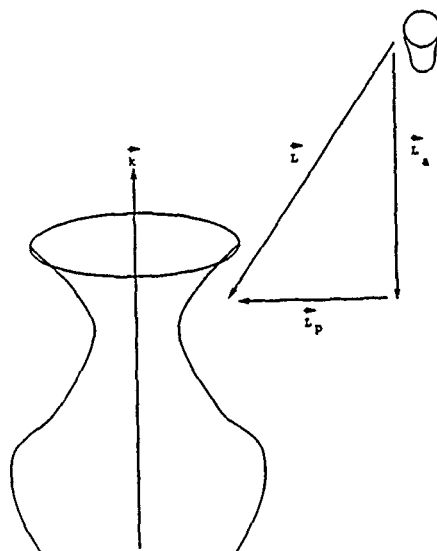


Figure 11. Imaging model for recovering the SHGC 3D axis translation: the point source directional vector \vec{L} is divided into orthogonal components, \vec{L}_a is parallel to the SHGC axis \vec{k} while \vec{L}_p is parallel to the SHGC cross-section plane.

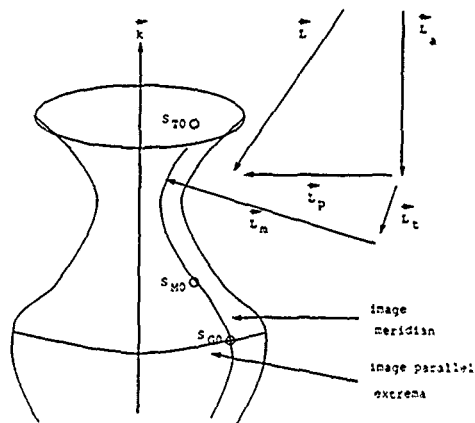


Figure 12. The \vec{L}_p component of source directional vector \vec{L} can be divided into orthogonal components \vec{L}_m and \vec{L}_t , which are, respectively, parallel and perpendicular to the surface normal at S_{GO} .

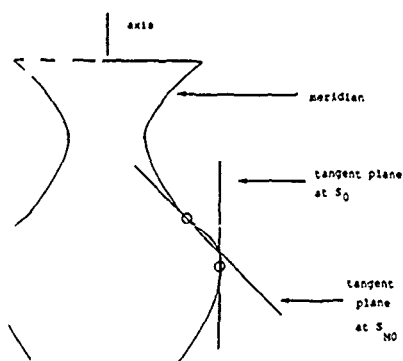


Figure 13. A side view of the SHGC meridian containing surface points S_{MO} and S_{GO} .

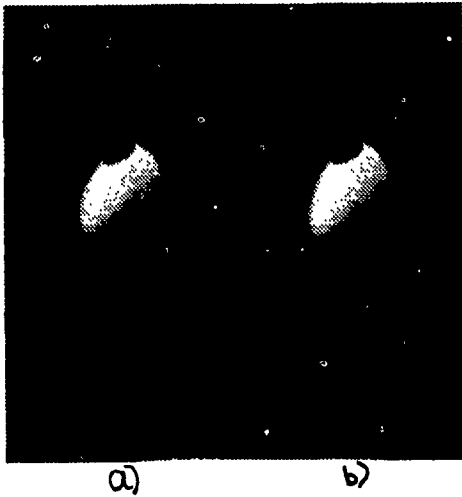


Figure 14. Recovering the slant: a. original SHGC; b. recovered SHGC.

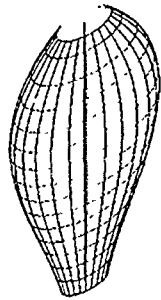


Figure 15. Ruled contour and image axis for SHGC of figure 14a.

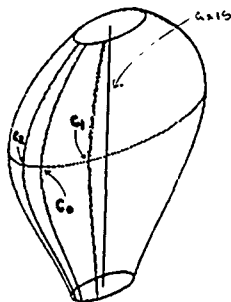


Figure 16. The three image points selected for recovering slant: C_0 having a value of $(0,0)$ in the gradient space (when image axis is vertically aligned) and C_1 and C_2 , whose tangents are at a 15 degree angle with respect to the tangent at C_0 .



Figure 17. Recovering the slant, a side view: a. original SHGC; b. recovered SHGC.

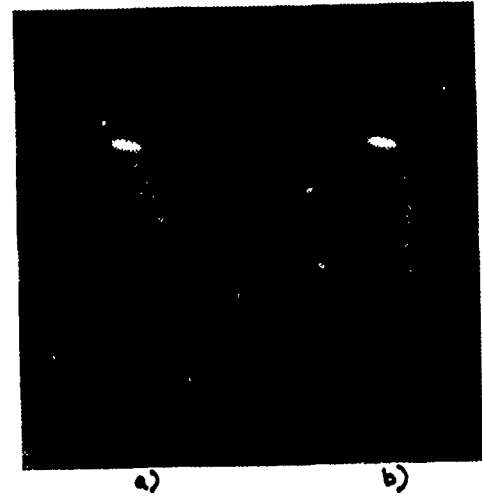


Figure 18. Recovering the translation: a. original SHGC; b. recovered SHGC.

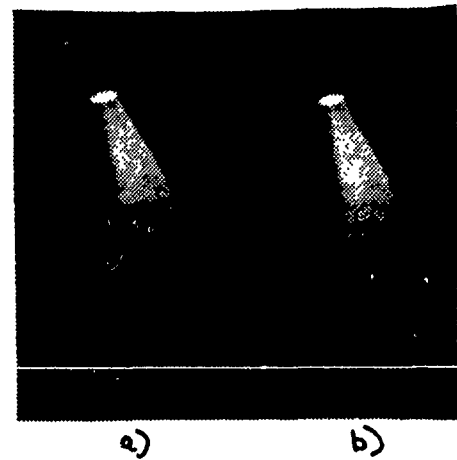


Figure 19. Recovering the translation, a side view: a. original SHGC; b. recovered SHGC.

Energy-Based Segmentation of Very Sparse Range Surfaces

Terrance E. Boulton and Mark Lerner
Department of Computer Science, Columbia University
NYC NY 10027.
tboulton@cs.columbia.edu lerner@cs.columbia.edu

Abstract

This paper describes a new segmentation technique for very sparse surfaces which is based on minimizing the energy of the surfaces in the scene. While it could be used in almost any system as part of surface reconstruction/model recovery, the algorithm is designed to be usable when the depth information is scattered and very sparse, as is generally the case with depth generated by stereo algorithms. We show results from a sequential algorithm that processes laser rangefinder data or synthetic data. We then discuss an implementation running on the parallel Connection Machine.

The idea of segmentation by energy minimization is not new. However, prior techniques have relied on discrete regularization or Markov random fields to model the surfaces to build smooth surfaces and detect depth edges. Both of the aforementioned techniques are ineffective at energy minimization for very sparse data. In addition, this method does not require edge detection and is thus also applicable when edge information is unreliable or unavailable. Our model is extremely general; it does not depend on a model of the surface shape but only on the energy for bending a surface. Thus the surfaces can grow in a more data-directed manner.

The technique presented herein models the surfaces with reproducing kernel-based splines which can be shown to solve a regularized surface reconstruction problem. From the functional form of these splines we derive computable bounds on the energy of a surface over a given finite region. The computation of the spline, and the corresponding surface representation are quite efficient for very sparse data. An interesting property of the algorithm is that it makes no attempt to determine segmentation boundaries; the algorithm can be viewed as a classification scheme which partitions the data into collections of points which are "from" the same surface. Among the significant advantages of the method is the capacity to process overlapping transparent surfaces, as well as surfaces with large occluded areas.

1 Segmentation: Introduction and Background

Segmentation is one of the most pervasive and most difficult problems in computer vision. It rears its ugly head in such subareas as: edge/region detection, motion detection, determination of textures, shape-from-X (for almost all X), calculation of disparity fields (stereo matching), model recovery, surface reconstruction and medical imaging. Unfortunately, the segmentation problem in each of these areas will not, in general, be solvable by the same techniques. One reason for the failure of the methods to extend to different the segmentation problems in the various subareas is because the assumptions about the data vary dramatically:

- In edge/region detection, the data is the intensity values of the image irradiance. The assumptions used for segmentation must be related to the process of image formation.
- In motion detection, the data can be either spatio-temporal intensity images or spatio-temporal surface information, and the segmentation assumptions must be related either to the flow of intensities as objects/self undergo motion (for spatio-temporal intensities images) or to object models and the physics of motion of said objects.

- In surface reconstruction, the assumptions used for segmentation must be related to models of world surfaces.
- In the recovery of disparity fields, assumptions must be tied to either a model of disparity fields, or a combination of models of world surfaces and the pair of image formation equations used to obtain the disparity field.
- etc. ...

Of these, the segmentation tasks in surface reconstruction, disparity field recovery and certain classes of motion detection problems, have been approached using segmentation coupled with recovery using an energy-based smoothness assumption, for example see [Terzopoulos 84], [Hoff and Ahuja 85], [Anandan and Weiss 85], [Blake and Zisserman 86] [Chou Brown 88], [Kanade 88]. In each of these cases, the attempt at segmentation can be roughly described as follows:

- Step 1 Do a initial smoothness-based reconstruction (this is generally a minimal energy surface or configuration).
- Step 2 Mark those parts of the reconstruction which are "not locally smooth" (generally with a gradient like operator) as discontinuities.
- Step 3 Adjust the reconstruction mechanism to deal with the newly marked discontinuities and go to Step 1.

Other segmentation approaches for surface reconstruction have incorporated smooth measures implied by volumetric models, for example, see [Pentland 86], [Bajcsy and Solina 87], [Rao, Nevatia and Medici] or local smoothness properties such as planarity or curvature consistency, see [Besel and Jain 86].

2 Motivation and a Different Formulation

This paper proposes a different model of segmentation which has some fundamental differences in the formulation of the problem. This section discusses this model, and also provides some motivation for it. In defining this model the section introduces the energy-based segmentation approach wherein the energy of reconstructed surface(s) is directly used to segment the data.

The traditional view of the surface segmentation problem is one of determining the "discontinuity boundaries" in surface depth, surface orientation and/or surface curvature. This approach usually requires some reconstruction of the surface, which is related to an *a priori* chosen measure of surface energy. Unfortunately, in order to correctly generate a surface reconstruction, knowledge of data segmentation is generally required. This results in a difficult chicken-and-egg problem. Thus, researchers may assume the scene consists of a specific class of surfaces (such as planar or convex). To make matters worse, the quality of the reconstruction in the neighborhood of an unmarked (i.e. as yet undetected) discontinuity is generally poor. Thus the localization of the discontinuity of iterative reconstruct/segment approaches, see e.g. [Terzopoulos 84] or [Hoff and Ahuja 87], will be questionable. Moreover, data from scenes with transparent surfaces cannot easily fit to these models.

A second shortcoming of the traditional approach is that it will require considerable post processing to handle extended multiply connected objects (say an object behind a picket fence) and may never be

able to handle transparent surfaces where locally there are only a few points on any one surface.

A final remark about traditional segmentation is related to the definition of "boundaries". It is well known that the perceived "boundaries" of surfaces in depth share many characteristics with subjective contours, see [Julesz 71], [Marr 81]. This suggests that a definition of "boundaries" in depth might be accomplished by some secondary processing which is shared with "boundary" detection from other visual modalities. Therefore the energy-based method of this paper can be combined with a secondary boundary-detection process to obtain both the shape and the outline of each surface.

To accommodate the above mentioned problems, this paper proposes that segmentation of 3D information should not attempt to determine boundaries. Rather, the segmentation should simply classify points as belonging to the same surface. The determination of boundaries will be relegated to some secondary process which is not discussed here. Of course this view cannot be taken too far, as there are limits both to the number of possible "transparent" surfaces and the number of times a background object is blocked by foreground objects. Psychological experiments have already demonstrated that such limits exist in the human visual system.

As mentioned above, what is desired is some measure which can be applied to groups of points to determine if they are part of the same physical surface. This section presents one such measure, surface bending energy, and discusses its appropriateness. The authors acknowledge that other measures might be used and the end of this section touches upon some of these alternatives.

Segmentation has been extensively studied in the context of image segmentation, and one might wonder if the algorithm herein is new / applicable to that domain. The crucial part of our algorithm is the use of "surface energy" to heuristically (though reliably) determine if two points are part of the same "extended region". Unfortunately, such measures have proven illusive for intensity images.

In computer vision, as well as other domains, researchers have used minimal surface bending energy as an assumption to aid in surface recovery, for example see [Grimson 81], [Franke 82], [Terzopoulos 84], [Wahba 84], [Hoff and Ahuja 85], [Choi and Kender 85], [Lee 85], [Blake and Zisserman 86], [Boulton 86]. Bending energy thus seems a natural choice for the "measure" to determine if a group of points belong to the same surface. The bending energy of a thin-plate surface f is given by:

$$\left\{ \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left(\left(\frac{\partial^2 f}{\partial x^2} \right)^2 + 2 \cdot \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 f}{\partial y^2} \right)^2 \right) dx dy \right\}^{\frac{1}{2}} \quad (1)$$

Of course, the use of bending energy can only be a partial basis for a practical measure for segmentation. Other issues that must be also be addressed include:

- What is the allowable class of functions for bending energy.
- What is the effect of surface size, or the area over which the energy is measured.
- What is the relationship between the energy, the number of data points and surface area.
- How to set the threshold for separation of a group, or alternatively to define the tradeoff between the number of surfaces and sum of the energies of these surface.
- When is the energy of a collection of points "too big".
- How does one locate the fewest number of "culprit" point(s) in the group, i.e., the credit assignment problem.
- What is the relationship between "depth" discontinuities and "orientation" discontinuities, and how do these characteristics effect the energy measure discontinuities.

3 An Energy-based Segmentation Algorithm

This section describes one way to realize an energy based segmentation algorithm given that one accepts the assumption of minimizing bending energy. However, the algorithm is easily modified to handle many related measures such as those described in [Boulton 86] and [Boulton 87]. The section describes the mathematical background of the algorithm.

The discussion of advantages and disadvantages of this approach are relegated to a separate section following this one.

The algorithm constructs initial approximations of the surfaces from the local data cluster 3-space*. These approximate surfaces are updated by subsequent processing. The algorithm then heuristically determines which point to add next (more below) and points are added to a surface as long as the addition does not cause the energy (see below) of said surface to exceed a certain threshold. If the surface cannot accept any remaining points, a "new" surface is created and the process repeats itself until all data points have been processed. A variation of the algorithm develops several surfaces in parallel by placing a point on several surfaces that can accept it without exceeding the threshold.

In its basic form, each pass of the algorithm computes the surface and corresponding energy that would result if each remaining point were separately added to the current surface. The system then adds the point which would cause the minimal rise in surface bending energy, assuming it does not exceed the specified threshold. Because of the monotonic and commutative nature of the energy measure, this approach will generally find the surface of minimal energy given the starting basis and the threshold, though it can be computationally expensive. This expense is being addressed in two ways. First, through a parallel implementation. Second, we recognize that the true minima is not always necessary, and we develop heuristics that develop the segmentation at lower cost.

The discussion of algorithmic details can be divided into seven smaller conceptual components which appear as separate subsections. These components are:

1. Definition of the model of world surfaces.
2. Definition of the reproducing kernel-based spline which is used to reconstruct the surfaces.
3. Definition of the energy measure.
4. Calculation of bounds on the energy of a reproducing kernel-based spline surface.
5. Heuristics used for (a) basis point selection, (b) point selection, and (c) culprit point selection, to remove some points and decrease the surface energy.
6. Method to merge similar surfaces into one surface.
7. A short discussion of the ongoing parallel implementation.

3.1 Definition of the Model of World Surfaces

The assumed model of world surface is intimately related to techniques for regularized surface reconstruction, see [Boulton 86]. An important set of these classes can be parameterized formally as those functions with their m^{th} derivative in H^η , where H^η is the Hilbert space of functions such that their tempered distributions ν in \mathbb{R}^2 have Fourier transform $\hat{\nu}$ that satisfy

$$\int \int_{\mathbb{R}^2} (|\tau|^{2\eta} \cdot |\hat{\nu}(\tau)|^2) d\tau < +\infty.$$

This class of functions, referred to as $D^m H^\eta$, is equipped with the m^{th} Sobolev semi-norm,

$$\| \cdot \|_{D^m} = \left\{ \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left(\sum_{i+j=m} \binom{m}{i} \left(\frac{\partial^m f}{\partial x^i \partial y^j} \right)^2 \right) dx dy \right\}^{\frac{1}{2}} \quad (2)$$

which, if $1 > \eta > 1-m$, results in a semi-Hilbert space. Note that if one chooses $m = 2, \eta = 0$, then using the properties of Fourier transforms, the above definitions yield exactly the space $D^2 L^2$ which was used by Grimson and Terzopoulos.

Let us now give an intuitive definition of these classes of functions. First, note that the spaces of "functions" assume the existence of the m^{th} derivative of the function, in the distributional sense. This means,

* The size of the cluster is user definable from 4-30 data-points. In general, the number must be at least 1 more than the size of the dimension of the null space of the energy measure. A number of heuristics have been developed to pick the basis points. These generally select an initial point along with a subset of its neighbors in 3 space that produce the lowest energy surface.

roughly, that the m^{th} derivative of the functions exists except on sets of measure of zero (e.g., at isolated points or lines). Then the classes $D^m H^0$, which are also known as $D^m L^2$, simply assume that the power of these functions is bounded. For the classes $D^m H^1$, $m > 0$ we have that the square of the spectrum of the derivatives goes to zero (as the frequency goes to ∞) faster than a specified polynomial of the frequency. Thus, these functions have less high frequencies and are "smoother" than functions which simply have m derivatives. For the classes $D^m H^1$, $m < 0$ we see that the spectrum of the derivatives is bounded away from zero, and that as the frequency goes to ∞ , the derivatives go to infinity no faster than a given polynomial. Thus, these functions have less low frequencies and are less "smooth" than most functions with m derivatives.

In the work reported here, we consider the class $D^2 H^{\frac{1}{2}}$ which, intuitively is the space of functions which are smooth (almost everywhere) up to derivatives of order approximately 1.5, i.e., they are significantly smoother than membrane surfaces but are not as smooth as thin-plate splines. The motivation for this choice of functions (i.e., this "intermediate" level of smoothing) is supported by the results of [Boult 87].

3.2 Definition of Reproducing Kernel-Based Spline

An essential ingredient of the current algorithm, at least from the point of view of efficient serial implementation, is the use of the reproducing kernel-based spline reconstruction as described in [Boult 86]. This section introduces some aspects of that algorithm necessary for later discussions.

We do not choose to interpolate the data, instead we follow the "regularization" approach of minimizing a smoothness term (the m^{th} Sobolev semi-norm) a weighted sum of squares of the distance of the surface from the data, i.e. we find the surface from our class of surfaces which minimizes:

$$\lambda \cdot \sum_{i=1}^n \frac{(\sigma(x_i, y_i) - z_i)^2}{\delta_i} + \|\sigma\|_{D^m}^2 \quad (3)$$

where the data z_i at point (x_i, y_i) , $i = 1, \dots, n$ is assumed to be on one surface. The *global smoothing parameter*, λ , should depend on the overall error in the initial data, and the factors δ_i allow for individual points to have greater "noise"; the factor λ effects the overall tradeoff between surface smoothness (as measured by the norm $\|\cdot\|_{D^m}$) and the fidelity to the data points z_i . The factor δ_i effects the contribution of a single data point so as not to penalize the surface as much (or to penalize it more, depending on the value of δ_i) for not closely approximating the data at that point. Techniques for choosing these parameters have been discussed by other researchers, see [Bates and Wahba 82].

One solution to the above reconstruction problem is a reproducing kernel-based spline. It can be shown, see [Meunet 83], that for the above model of world surfaces, $D^2 H^{\frac{1}{2}}$, the appropriate reproducing kernel here is

$$K(x, y; u, v) = \gamma((x - u)^2 + (y - v)^2)^{\frac{3}{2}}$$

for some constant γ

Given the above kernel, the spline (i.e., the reconstruction of the $2\frac{1}{2}D$ sketch) which approximates the data

$$z = z_1, \dots, z_k = \{f(x_1, y_1), f(x_2, y_2), \dots, f(x_k, y_k), \quad i = 1, \dots, k\}$$

can be developed as:

$$\sigma_z = \sum_{i=1}^k \alpha_i K(x, y; x_i, y_i) + \sum_{i=1}^3 \beta_i p_i(x, y) \quad (4)$$

where $p_1(x, y) = 1$, $p_2(x, y) = x$, $p_3(x, y) = y$. The constants α_i and β_i are the solution to the system of linear equations:

$$\begin{array}{cccccc} \alpha_{1,1} & \dots & \alpha_{1,k} & \beta_{1,1} & \beta_{1,2} & \beta_{1,3} & z_1 \\ \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_{k,1} & \dots & \alpha_{k,k} & \beta_{k,1} & \beta_{k,2} & \beta_{k,3} & z_k \\ \hline c_{1,1} & \dots & c_{1,k} & d_{1,1} & d_{1,2} & d_{1,3} & 0 \\ c_{2,1} & \dots & c_{2,k} & d_{2,1} & d_{2,2} & d_{2,3} & 0 \\ c_{3,1} & \dots & c_{3,k} & d_{3,1} & d_{3,2} & d_{3,3} & 0 \end{array}$$

where

$$\begin{aligned} \alpha_{i,j} &= \delta_i / (\alpha_i \lambda 8\pi) \quad i = 1, \dots, k; \\ \alpha_{i,j} &= \alpha_i (K(x_j, y_j; x_i, y_i)), \quad i = 1, \dots, k \quad j = 1, \dots, k \quad i \neq j; \\ \beta_{i,j} &= c_{j,i} = \beta_j p_j(x_i, y_i) \quad i = 1, \dots, k, \quad j = 1, \dots, 3; \\ \text{and} \quad \mathcal{D}_{i,j} &= 0 \quad i = 1, \dots, 3 \quad j = 1, \dots, 3; \end{aligned} \quad (5)$$

The important properties of the above solution to the surface reconstruction problem are:

1. The algorithm is efficient for very sparse data (anything more than 3 non-collinear points will do, and the fewer the number of points, the faster the surface can be computed).
2. The surface is defined by the solution to a linear system which depends only on the location of the data and the certainty of the observations. The certainty is used to control the amount of smoothing, with less smoothing for less noisy sensors. If the solution to this system can be updated quickly, the surface can also be updated quickly.
3. The surface is given in a functional form, thus the evaluation of derivatives is trivial, and bounds on the energy of the surface can be computed analytically.
4. The surfaces are independent of the "boundaries" of discontinuities, and depend only on the data values. However, the actual surface will change if the number/value of data points on the boundary are changed.
5. The definition of the nullspace can be changed to consider different polynomial combinations of initial data values.

3.2.1 Short discussion of the updatable QR algorithm This section briefly discusses the way the algorithm updates the linear system to allow for an efficient update of the surface for serial implementations. The algorithm begins by doing a QR decomposition of the initial linear system. Then, using Givens rotations, the algorithm can allow for the addition or deletion of any row/column (in fact, it can handle any rank one modification). We make use of the capacity to delete data from the system without recomputing the linear system in order to remove culprit points, as will be discussed later.

The computation of the initial QR decomposition for k data points requires time $O(\frac{1}{3}k^3)^1$. The addition/deletion then costs $O(k^2)$, and the recomputation of the solution with the new system costs $O(k^2)$. The algorithm is numerically stable which is important since the condition number for reproducing kernel-based spline problems can be moderately large.

A secondary advantage of using the QR decomposition is that changing the values of the data (while leaving the position unchanged) requires only $O(k^2)$ to recompute the solution. We exploit this for merging surfaces by interpolating data onto a fixed grid with a pre-computed solution. While these algorithm are not new, the authors hope that this brief disclosure will alert the vision community to its potential uses. The algorithm is precisely defined in [Daniel et al. 76] and widely available implementations can be found in various mathematical libraries, e.g. IMSL.

3.3 Definition of the Energy Measure

The basic form of the energy measure is given by equation 2 except that the region of integration may be different than that expressed therein.

The energy of the surface will depend on the size of the region in \mathbb{R}^2 over which the energy norm is computed. The two most natural choices are \mathbb{R}^2 itself, and the convex hull of the data defining the "current" surface. Unfortunately, neither is appropriate. For the class used in the initial tests, the integral over \mathbb{R}^2 is not necessarily finite.¹ While the energy norm over the convex hull of the data defining the "current" surface is obviously finite,² this choice has two difficulties:

1. The convex hull would continuously change as new data points were added to a surface.

¹While the decomposition could in fact be computed by adding every row one at a time, this doubles the cost of the decomposition, and may slightly effect the quality of the solution.

2. The use of a domain which ends near the data points will allow the addition of new points to lower the surface energy, thus the energy will no longer be monotonically increasing, making a region growing method less stable.¹¹

3.4 Derivation of Bounds on Energy of Surface

Given the definition of the spline as in equation 4, one can symbolically compute bounds on the energy. To begin, the exact form of the energy integral is manipulated to explicitly expand the squaring operation and move the differentiation and integration inside the sum, to wit:

$$\| \sigma \|_{D^2} = \left\{ \int_{X_1}^{X_n} \int_{Y_1}^{Y_n} \left(\left(\frac{\partial^2 \sum_{i=1}^k \alpha_i K(x, y; x_i, y_i)}{\partial x^2} \right)^2 + 2 \cdot \left(\frac{\partial^2 \sum_{i=1}^k \alpha_i K(x, y; x_i, y_i)}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 \sum_{i=1}^k \alpha_i K(x, y; x_i, y_i)}{\partial y^2} \right)^2 \right) dx dy \right\}^{\frac{1}{2}} \quad (6)$$

By letting the integration be over the square $[-B, B]^2$ we have can obtain

$$\| \sigma \|_{D^2} = \left\{ \sum_{i=1}^k \sum_{j=1}^k \alpha_i \cdot \alpha_j \cdot \left(\int_{-B}^B \int_{-B}^B (K_{xx}(x, y; x_i, y_i) \cdot K_{xx}(x, y; x_j, y_j) + 2 \cdot K_{xy}(x, y; x_i, y_i) \cdot K_{xy}(x, y; x_j, y_j) + K_{yy}(x, y; x_i, y_i) \cdot K_{yy}(x, y; x_j, y_j)) dx dy \right) \right\}^{\frac{1}{2}} \quad (7)$$

While it would be most appropriate to integrate the terms symbolically in the last of the above equations, the authors (and MACSYMA) have been unable to obtain a solution. Fortunately, a symbolic solution can be obtained for the upper bound using the fact that we can compute:

$$I(B, s, t) = \int_{-B}^B \int_{-B}^B \{ K_{xx}(x, y; s, t)^2 + 2 \cdot K_{xy}(x, y; s, t) + K_{yy}(x, y; s, t)^2 \} dx dy \quad (8)$$

While we will only report on the classes $D^2 H^{\frac{1}{2}}$ and $D^3 H^{\frac{1}{2}}$, we have obtained closed form solutions for $I(B, s, t)$ for various values of m and n . These are given in the following table:

$D^2 H^{\frac{1}{2}}$	$180B^2(t^2 + s^2) + 120B^4$
$D^3 H^{\frac{1}{2}}$	$252B^2$
$D^2 H^{\frac{1}{2}}$	$\frac{1}{2} \cdot (11900B^2t^6 + (35700B^2s^2 + 71400B^2)t^4 + (35700B^2s^4 + 142800B^4s^2 + 66640B^6)t^2 + 11900B^2s^6 + 71400B^4s^4 + 66640B^6s^2 + 8160B^8)$
$D^3 H^{\frac{1}{2}}$	$17100B^2(s^4 + t^4) + 34200B^2t^2s^2 + 45600B^4(s^2 + t^2) + 10640B^6$
$D^4 H^{\frac{1}{2}}$	$87300B^2(s^2 + t^2) + 58200B^4$

Table 1. Table of $I(B, s, t)$ for different classes of functions.

¹¹If one considers the class $D^2 H^0$, then the energy norm is, by definition, finite. However, the energy value may be very large and thus may be numerically unstable.

¹²Intuitively one can assume a Lebesgue integral which may ignore sets of measure zero. More formally, the definition of the class was in terms of distributions and the energy measure can be defined in a distributional sense as well.

¹³Intuitively this anomaly can be understood by noting the following:

- Surface energy measures the "bending" of the surface in the domain of integration.
- The definition of the class of surfaces insures that the value of "the surface" must go zero as the point of evaluation approaches infinity, thus outside the convex hull of the data the surface will approach zero, and this may cause some "ringing" in the surface if the data density is "low" near the edge of the region.
- By increasing the data density, the value of the surface inside a bounded region can be forced to approach a planar surface which has zero energy. Thus after initially building a surface using points on some region boundary, adding new data points within the region can "push" the high energy portions (i.e. bent areas) outside the convex hull. This results in a nonmonotonic energy measure.

The choice of norm is significant. The second norm gives zero energy to planar surface but large energy to conic sections. Thus it is appropriate when the world model is known to be mostly planar. This is exactly the case for many man-made objects, such as bent sheet metal, desks, and other rectangular scenes. In contrast, the third norm gives zero energy to conic sections, and thus it is appropriate when many round objects are in the scene.

Once the norm is selected we set the maximum energy thresholds according to the precomputed energy for prototype shapes in the scenes. A number of intermediate thresholds are also used to slowly increase the surface energy. This prevents the erroneous segmentations that occur if only the final threshold values occur, since it bends the surface slowly. The algorithm therefore places points onto the surface where they most easily fit. The next larger threshold is used when no more points can be placed within the current one.

3.5 Heuristics

Heuristics reduce the cost of the energy-based method by selecting points to be considered for the various surfaces, and also for deleting points from these surfaces. Good heuristics should focus the system by selecting the points that should be added to each surface. The points selected by a heuristic can then be tested for the minimum energy criteria, and only the low-energy points are used. We now discuss several subproblems where heuristics help, in particular building the initial surfaces, adding points to them, and merging or eliminating duplicate surfaces.

The system combines two kinds of heuristics. The domain independent heuristics described here are sufficiently general to provide reasonable results for many scenes, including scenes with occluding surfaces, low-energy surfaces and high-energy surfaces. Domain specific heuristics increase efficiency by selecting points according to the expected scene characteristics, for example if information is available on the curvature of a surface the program can construct an initial solution from the points that are "likely" to be on the surface. This provides a good basis for adding more points by the energy-based method. Domain specific information also can be used to set the thresholds for the surfaces.

Construction of the initial "basis" surfaces. Each surface is initially described by a small number of points, between 4 and 30. Good starting surfaces are essential for accurate segmentation, and therefore the program must find points that all lie on the same actual surface. This is done in two steps. First a significant single point is found as a "seed" for the basis. Second, a subset of its neighbors in 3-space are selected. Ideally each "seed" point should be from a different surface. Since we cannot achieve this without knowing the segmentation, we instead select the seed according to the following heuristics. A physical explanation of why it is a good heuristic follows each one:

- Pick the nearest point (physically, this is a point on the closest object).
- Pick the farthest point (physically, a point on the most distant object).
- Pick randomly, but not near any seed value that has already been chosen.
- Pick a point near to a specific XY coordinate on a grid (appropriate when approximate object locations or distributions are known).
- Pick the point that is farthest from one picked previously (span as much of the scene as possible).

The lowest energy surface is then built from the seed and a subset of the neighboring points. This entails a combinatorial search of the M near neighbors for the lowest energy N element surface. The starting surfaces are identified by selecting an "interesting" point and its neighbors in 3-space. We then perform an exhaustive search of the M nearest neighbors to build N -point surfaces. If M is too small the constructed surface may span several actual surfaces. Likewise, if N is too small the resulting surface will not be sufficiently descriptive to permit accurate selection of points. The combinatorial search is expensive, and is an excellent candidate for parallel processing on a large data-parallel machine.

It is important to restrict the search to the near neighbors because it is otherwise quite easy to build large nearly-planar surfaces that cut through several surfaces. Moreover, the points must not be too close to each other because the resulting reconstruction would be highly distorted by surface noise. Otherwise the points are not distinguishable in XY coordinates but the resulting surface can have huge energy due to the variation in Z , which will be quite relative to the difference in XY values. The Z values could be due to noise or to multiple surfaces.

The algorithm can recover from incorrect starting surfaces. The smoothness assumption identifies as erroneous any surface that has a large energy with a small number of points, and such surfaces can be pruned and the points reused on other surfaces. Secondly our system periodically deletes points and merges surfaces. These actions allow a surface to shed points that were erroneously placed onto it, and then to combine similar surfaces into one denser surface.

Point selection: what point to add? Once the initial basis surface has been constructed, points must be selected and then added to the surface which can accept them with minimum increase in energy. The exhaustive method would require N^2m update operations (for N points and m surfaces) at a cost of N^2 per update. This N^4 cost is excessive, and therefore we first estimate the energy. We have been successful with an estimation method that considers only the points which are near to some point already on the surface (we will describe this shortly). The nearness criteria also prevents the erroneous addition of a non-surface point that just "happens" to fit onto a partially developed surface, for example with occluding transparent surfaces.

A point is near a surface if the distance to some point on the surface is less than half the diagonal of the box that bounds the points already on the surface. The point is temporarily ignored if it is too far from the surface, and is considered at a later iteration after the surface has accepted closer points (and the bounding box is larger). The near points are then ranked by a weighted distance formula. This combines the distance to the nearest point already on the surface, with the proximal distance from the point to the partially reconstructed surface. The points are then added in order of increasing cost subject to a global threshold.

Culprit identification: when and how to pick points to delete? Misclassifications of data during the incremental addition of points to surfaces results in excessively large surface energy. Indeed, our underlying assumption is that smooth surfaces have low energy. If the surface is not smooth it becomes important to delete some point(s) from the surface. In general the point which makes the largest contribution to the energy does not belong to the surface and is the classification error. Thus it should be deleted. The exhaustive method to find this point (i.e. try each possible point) is uneconomical. Therefore we need some way to predict which point is the "culprit" responsible for the excess energy.

We have two methods to delete points. The first method prunes away any data item that is more than three standard deviations away from the surface reconstruction, i.e. the reproducing kernel-based spline. After segmentation is complete at each threshold, the program computes the standard deviation of the distance from the data to the reconstructed surface. The distant points are deleted, and the test is reapplied until the change in standard deviation is less than 5 percent. Note that the updatable QR algorithm allows deletion of arbitrary rows and columns, so this delete operation is fairly economical. We are very pleased with the culprit deletions, and note that our results are free from any "spikes" that indicate misclassification.

A second indicator of the "culprit" is the value of α in the linear system which recovered the spline parameters. Intuitively this is because a large α makes a large contribution to the energy. We use culprit deletion by removing the point with largest α after every 10 insertions. If a point is erroneously removed it can still be added back in a subsequent add operation. This is because the increase in energy at each step is a good predictor of the correct segmentation. We have shown experimentally that an incorrect data point is associated with a large alpha value, and are studying this method further.

Segmentation: when to create a new surface? We have investigated two methods: first of completely building one surface at a time, and second of building all surfaces simultaneously by gradually increasing

the global energy threshold. The second method gives significantly better results than the first, since it completes all low-energy classifications before it increases the energy. The first method can misclassify when a point can be on either S_1 with large energy or on S_2 with lower energy. The approach of building all surfaces may also improve performance of the parallel implementation. However, the simultaneous building of multiple surfaces will construct redundant surfaces, and also place co-surface points onto different reconstructions. This problem is solved by merging similar solutions as described below.

Merging: when and how to combine surfaces into one? Merging similar surfaces is important for two reasons. First, we do not know of any way to guarantee that the initial basis surfaces are from different actual surfaces. Therefore we necessarily construct multiple similar surfaces. These surfaces may overlap, and thus they cannot simply be merged at their boundaries. Secondly, we anticipate increases in processing speed in the parallel version by simultaneously processing several subsamples of the data in different sections of the parallel computer, and then merging these results into one representation. In both cases the merge operation should construct one new smooth surface from the data of both similar surfaces.

We have two definitions of "closeness". The first minimizes the sum squared distance between the reconstruction of surface S_1 with the data of surface S_2 at those points which are within the bounding box of S_1 (and likewise the data of S_1 with the surface of S_2). The purpose of evaluating the sum only within the bounding box (i.e. between the minimum and maximum $X - Y$ values) is to prevent extrapolation error.

The second way we test if two surfaces are sufficiently "close" to be merged is by interpolating the surfaces S_1 and S_2 on a sparse and fixed $X - Y$ grid, and then constructing a new surface from these samples. We use precomputation to efficiently compute the energy of such a system, which reduces the computational expense to simple data interpolation (at the grid values) and back-substitution. This is much cheaper than building a new surface by adding each point to it.

A surface is considered for merging only if it has at least the median number of data values on it. The merge cost of all such pairs is computed, and the merges are performed in order of increasing energy. Each point of the less-dense surface is considered for the denser surface, and is added if the point does not exceed k times the median energy of the less-dense surface. In this manner erroneous points will not be merged, but instead are returned to the pool of unprocessed data and will be placed onto a surface in a subsequent iteration.

Pruning: when should a surface be eliminated? Sometimes a surface is incorrectly started from the data on several different scene surfaces. The energy of such surfaces is generally high, and thus the surface does not accept many additional data points. These sparse surfaces are discarded and the points are reused on some other surface. The program thereby recovers from false starts by resegmenting data that does not produce a surface.

3.6 Parallel Implementation

An prototype of segmentation is operational on the massively parallel Connection Machine. The CM-2 has 65,536 processors with 512 megabytes of memory and a 300 gigabyte/sec memory bandwidth. This gives 9K bytes RAM per processor. Although the processors are small 1-bit PEs, the typical aggregate operation speed is 2500 mflops for double precision on a 4Kx4K matrix multiplication, and 5000 mflops for a dot product. For a complete technical summary see [Thinking Machines].

We have implemented for the CM-2 the updatable QR algorithm for solving linear systems, construction of the linear system (equation 5), evaluation of surface energy in parallel, and adding points to a surface. This software has processed data from a number of synthetic surfaces including spheres and planes. The software is written in the *Lisp language.

We use data parallelism to accelerate the combinatorial search for the initial basis surfaces. The algorithm distributes a different subset of the data to each processor. In one data-parallel step each processing element simultaneously constructs the minimum energy surface that fits its subset. The lowest energy solutions are retained as the starting-bases that accept additional data points.

In the second form of parallelism, algorithm parallelism, many processors cooperate to solve a single problem instance. This form of parallelism is appropriate when there are more processors than problem instances. In this case many processors may cooperate to solve one problem. For example, both the recovery of the spline parameters and surface merging may be done in parallel. We have developed a parallel version of the updatable QR algorithm [Daniel et al. 76] and are currently investigating ways to improve its efficiency. The quick solution of linear systems will allow us to process more data than we can on a uniprocessor. This should allow us to process thousands of points, as occurs in non-sparse data, and also to process many surfaces. The heuristics can also be processed in parallel, in particular the local properties of all points can be explored simultaneously.

4 The pluses and minuses

This section critically reviews the algorithm described in this paper pointing out some of the major advantages +, major problems -, and some aspects which can be viewed as either a pro or a con \pm .

- + The segmentation process is based on surfaces having low bending energy, a heuristic which can be directly related to the physical process of surface formation. Since this heuristic is often used in surface reconstruction, it is a natural for segmentation as well. Because we do not attempt to detect discontinuities, the algorithm is well behaved for very sparse data, and even handles transparent and occluded objects with few problems.
- + The functional form of the reproducing kernel-based spline allows for analytic computation of bounds on the surface energy, thus making the segmentation process reasonably computationally efficient. The functional form of the energy bounds are quite simple.
- + While not presented here, the algorithm can easily be extended to handle the case of derivative information (e.g. surface orientation or curvature) in addition to depth data. The extension is accomplished by allowing a more complex surface reconstruction scheme, see [Kender Lee Boulton 85].
- \pm The complexity of the algorithm, with the selection heuristic of global minimal energy addition, is $O(n^4)$ for n points, and with the other heuristics it is $O(n^3)$. For very sparse data, this is a significant saving over discrete regularization costs, but as the data densities grow, the algorithm becomes too costly.
- \pm The algorithm does not recover "boundaries" for the segmented data. This is advantageous because it allows for transparent and/or occluding surfaces, and because data is generally sparse (and often noisier) near the boundary resulting in a poor boundary definition. This is a disadvantage because it requires a secondary processes (possibly using ideas borrowed from work on subjective contour perception or Gestalt psychology) to determine the actual boundary.
- \pm The algorithm can easily be adapted to different measures of surface smoothness. This is advantageous because it allows for greater flexibility, but disadvantageous because determination of the most appropriate measure is difficult. The measure used in the experiments presented herein has proved to be a reasonable one.
- \pm The algorithm is based on reproducing kernel-based splines which are essentially a global surface reconstruction algorithm and provide for efficient serial implementation for sparse data (say < 500 points per surface on a 512×512 grid). If there are more points, the algorithm can be extended to use local reproducing kernel-based splines (loosely based on [Franke 82]), at the cost of making the surface definition localized to patches.
- \pm The reproducing kernel-based spline with updates to the QR factorization provide an efficient serial implementation. However, they do not suggest an efficient parallel implementation other than the trivial extension of doing the matrix computations in parallel. Future work may explore the possibility of using parallel multi-grid techniques to solve a discretized version of the surface reconstruction problem which can then be used in the segmentation algorithm. Unfortunately how one evaluates energy in this case is still unclear.

- \pm The order of processing of points currently effects the resultant segmentation in all but the exhaustive heuristic. This is especially true when two surfaces come into direct contact and join in a rather smooth fashion (e.g. a wedge). This may actually be used to help in the segmentation process by processing the data in multiple orders and using any difference in data labeling to suggest a refined segmentation.
- The algorithm currently uses a threshold on the energy bounds for surfaces in the scene. A schedule of surface thresholds is currently manually set, and future work (in progress) attempts to redress this issue. Luckily, this threshold for energy-based segmentation does not seem too sensitive as say thresholds for segmentation of an image based on intensity, e.g. (variations of 10% of the threshold are generally indistinguishable).
- The algorithm assumes one is interested in smooth surfaces and will most likely fail when this assumption is not satisfied. Unfortunately, the algorithm cannot even determine if the assumptions are satisfied (For example, consider a rough surface similar to a plane covered with a large number of small densely packed cones. If the data supplied to the algorithm are points on the background and the peak values of the cones, the algorithm is hopelessly doomed to predict two planar surfaces.)
- The algorithm is surface based, and cannot deal with data from multiple views of a volumetric object. Additionally, it will often fail if noise is such that a single x, y location is assigned multiple data values (of the same type).

5 Experimentation

This section describes some of the initial experimentation with the segmentation algorithm. The reader should remember that the experimentation involves some human interaction (to determine thresholds) and most of the examples highlight the system's best behavior. The experiments we run on a Sun Sparc1 workstation configured with 12M of memory.

We show here the results from two scenes. The first is *synthetic* data for three overlapping and intersecting superquadric surfaces with very similar shapes. The results show excellent differentiation between the surfaces. Segmentation results are shown as "needle-plots" where the length of the line is the distance from the $X - Y$ plane. We show the segmentation of ten segments the was computed by increasing the energy four times. Next we show the segmentation into only 5 segments after 5 merge cycles, followed by the reconstructed surfaces. Note there are no erroneous segmentations and thus the output is free of spikes.

We have processed laser range finder data from the Purdue Vision Lab, provided by Avi Kak, and also laser data from the University of Utah. We show only one of these more difficult data sets here that consists of complex round shapes.

6 Conclusions and future work

This paper has presented a new algorithm for segmentation of depth data. The algorithm is based on adding points to a surface only when doing so does not increase the surface bending energy above a user determined threshold. The algorithm has been experimentally tested and in most cases correctly labels all data points. The algorithm does not determine boundaries between segmented surfaces, and this which allows it to handle extended objects occluded by other objects and transparent objects.

One of the most obvious failings of the approach is the current dependence on a global thresholding technique to realize the segmentation. Such a process is doomed to be troublesome unless a systematic determination of the threshold is possible. Future work will address this issue and will also investigate the use of adaptive thresholding (depending on the actual data) and the use of other properties, say rate of change of energy, as the means of realizing segmentation.

The algorithm as presented has little theoretical basis for the use of energy as an indicator for segmentation. Of course, in a worst case setting, segmentation is an unsolvable problem. However, on the average there is still hope of determining a theoretical basis for segmentation. Future work will explore this idea, and attempt to show that

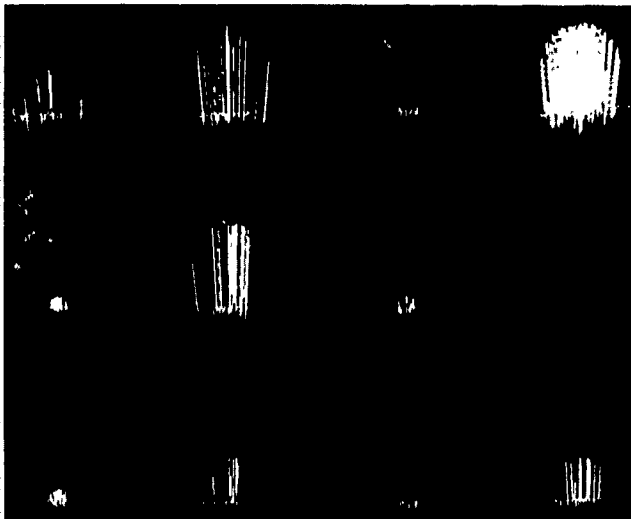


Figure 1: This shows a needle plot of three overlapping spheres, after 4 segmentation cycles at increasing energy values. The data was generated with radius and center as follows. The upper right shows the unsegmented data. The recovered segments are listed to the left. The parameters of the spheres are. Top surface, $r = 0.8$, $c = (0.0, 0.0, 1.0)$. Middle surface, $r = 0.25$, $c = (0, 0, 0)$. Bottom surface: $r = 0.25$, $c = (-0.5, -0.5, -0.5)$. Each surface is sampled at 150 points on a spherical-coordinate system and randomly shifted from the grid. Uniform random noise in the range $[-0.05, 0.05]$ was added to the z values of the data.

Segments 0, 1, 6, 7 and 8 represent the smallest sphere. Note there are no Z classification errors, though there is one error in the X direction of surface 6. Segments 3 and 9 represent the middle sphere, and segments 4 and 5 represent the largest sphere.

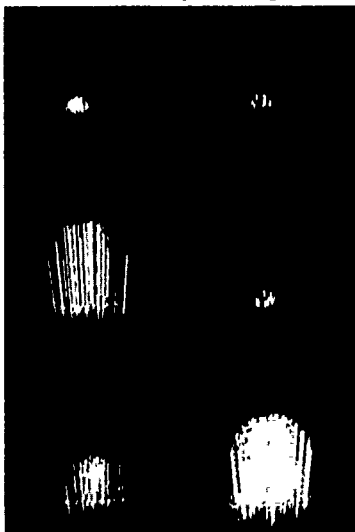


Figure 2: This shows the segmentation of the synthetic data after 5 merge cycles. The merged segments are (0, 1) (2, 3, 9), (4, 5) and (6, 7). Segments 1, 7 and 8 are not yet merged into one surface.

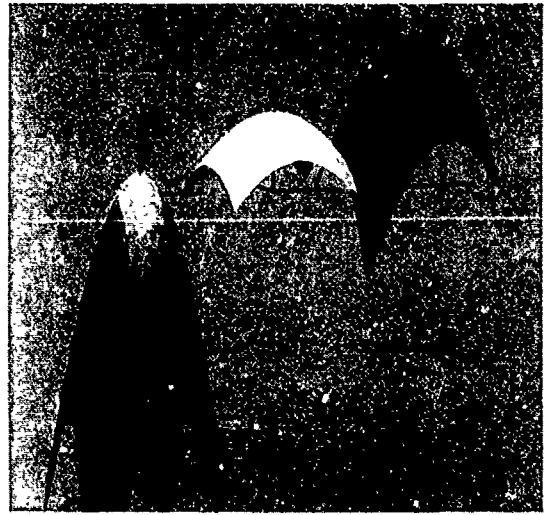


Figure 3: This shows all the recovered surfaces from the above example.

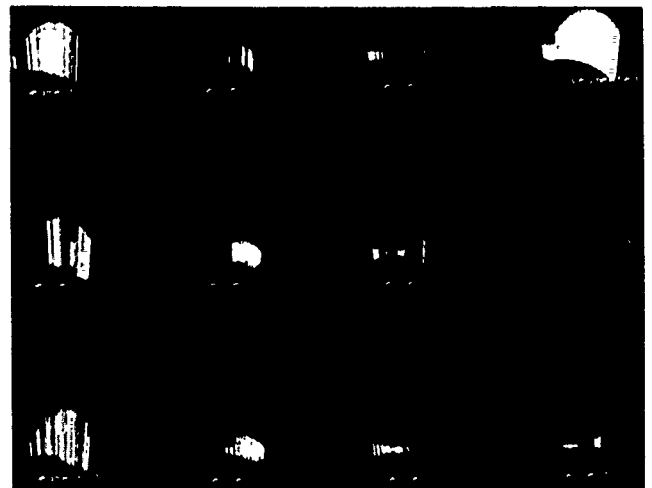


Figure 4: This shows laser range finder data for three-round surfaces (Purdue-Vision Lab data). The unsegmented data is in the upper right, and the segments are to the left.



Figure 5: This shows a partially merged version of the three round surfaces. Segments (0, 1) have been combined, as have segments (5, 3, 9) and (6, 7, 8).

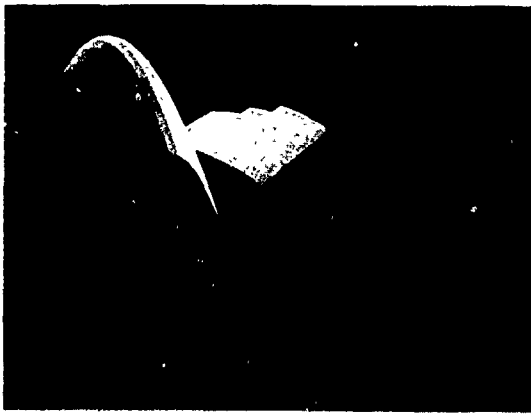


Figure 6:

This shows the reconstruction and segmentation of the laser data for three round surfaces culprits elimination. The graphic rendering shows patches only in the regions where data values were actually found

(Conclusions and future work, continued)

surface energy can be related to the probability of all data-points being on the same surface. This will likely borrow from the work of [Kimeldorf and Wahba 70], where a fundamental relationship is shown between reproducing kernel-based splines and optimal Bayesian estimators.

Acknowledgments

This work was supported in part by Darpa Contract #N00039-84-C-0165 and in part by the National Science Foundation Grant #IRI8800370. Thanks to Cliff Beshers for his help with the graphics displays

References

- [Allen 85] P. K. Allen. *Object Recognition Using Vision and Touch*. PhD thesis, University of Pennsylvania, Department of Computer Science., 1985.
- [Anandan and Weiss 85] P. Anandan and R. Weiss. Introducing a smoothness constraint in a matching approach for the computation of displacement fields. In *Proceedings of the DARPA Image Understanding Workshop*, pages 186-195, DARPA, 1985.
- [Bajcsy and Solina 87] R. Bajcsy and P. Solina. Three dimensional object representation revisited. In *Proceedings of the IEEE Computer Society International Conference on Computer Vision*, pages 231-240, IEEE, June 1987.
- [Bates and Wahba 82] D. Bates and G. Wahba. Computational methods for generalized cross-validation with large data sets. In C.T.H. Baker and G.F. Miller, editors, *Treatment of Integral Equations by Numerical Methods*, pages 283-296, Academic Press, New York, 1982.
- [Besel and Jain 86] P. J. Besel and R. C. Jain. Segmentation through symbolic surface description. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 77-85, IEEE, 1986.
- [Blake and Zisserman 86] A. Blake and A. Zisserman. Invariant surface reconstruction using weak continuity constraints. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 62-68, IEEE, 1986.
- [Boult 86] Terrance E. Boult. *Information Based Complexity in Non-Linear Equations and Computer Vision*. PhD thesis, Department of Computer Science, Columbia University, 1986.
- [Boult 87] T.E. Boult. What is regular in regularization? In *Proceedings of the IEEE Computer Society International Conference on Computer Vision*, pages 457-462, IEEE, June 1987.
- [Choi and Kender 85] D.J. Choi and J. R. Kender. Solving the depth interpolation problem with adaptive chebyshev acceleration method on a parallel computer. In *Proceedings of the DARPA Image Understanding Workshop*, pages 219-223, DARPA, 1985.
- [Chou Brown 88] Chou P., C. M. Brown. The theory and practice of bayesian image labeling. In *International Journal of Computer Vision* (to appear).
- [Daniel et al. 76] J.W. Daniel, W.B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization. *Math. Comp.*, 30:722-795, 1976.
- [Franke 82] R. Franke. Smooth interpolation of scattered data by local thin plate splines. *Comp. & Math. with Applications*, 8(4):273-281, 1982.
- [Grimson 81] W. E. L. Grimson. *From Images to Surfaces. A Computational Study of the Human Visual System*. MIT Press, Cambridge, MA, 1981.
- [Hoff and Ahuja 85] W. Hoff and N. Ahuja. Surfaces from stereo. In *Proceedings of the DARPA Image Understanding Workshop*, pages 98-106, DARPA, 1985.
- [Hoff and Ahuja 87] W. Hoff and N. Ahuja. Extracting surfaces from stereo images: an integrated approach. In *Proceedings of the IEEE Computer Society International Conference on Computer Vision*, pages 284-294, IEEE, 1987.
- [Julesz 71] B. Julesz. *Foundations of Cyclopean Perception*. University of Chicago Press, Chicago, IL, 1971.
- [Kanade 88] Matthies L., R. Szeliski, T. Kanade. Incremental estimation of dense depth maps from image sequences. In *Proceedings of Computer Vision and Pattern Recognition 1988*, pages 366-374.
- [Kender Lee Boult 85] Information-based complexity applied to optimal recovery of the 2 1/2 D sketch. In *Proceedings of the IEEE Computer Society Workshop on Computer Vision. Representation and Control*, pages 157-167, IEEE, October 1985.
- [Kimeldorf and Wahba 70] G.S. Kimeldorf and G. Wahba. A correspondence between bayesian estimation on stochastic processes and smoothing by splines. *Annals of Math. Stat.*, 11(2) 495-502, 1970.
- [Lee 85] D. Lee. *Contributions to Information-based Complexity, Image Understanding, and Logic Circuit Design*. PhD thesis, Department of Computer Science, Columbia University, 1985.
- [Marr 81] D. Marr. *Vision. A Computational investigation into the human representation and processing of visual information*. W.H. Freeman and Company, San Francisco, 1981.
- [Meinguet 83] J. Meinguet. Surface spline interpolation: basic theory and computational aspects. *Institut de Mathematique Pure et Appliquee, Universite Catholique de Louvain*, 35, 1983.
- [Pentland 86] A. Pentland. *Recognition by Parts*. Technical Report 406, SRI International, December 1986.
- [Rao, Nevatia and Medioni 87] K. Rao, R. Nevatia, and G. Medioni. Issues in shape description and an approach for working with sparse data. In *Proceedings of the AAAI Workshop on Spatial Reasoning and Multi-sensor Fusion*, pages 168-177, St. Charles IL, October 1987.
- [Terzopoulos 84] D. Terzopoulos. *Multiresolution Computation of Visible-Surface Representations*. PhD thesis, MIT, 1984.
- [Thinking Machines] Thinking Machines. Connection Machine (K) Model CM-2 Technical Summary. Thinking Machines Incorporated, Cambridge Mass., 1987.
- [Utah 85] Utah. *The University of Utah range database manual*. Technical Report, University of Utah, 1985.
- [Wahba 84] G. Wahba. Surface fitting with scattered noisy data on euclidean d-space and on the sphere. *Rocky Mountain Journal of Mathematics*, 14(1):281-299, 1984.

Straight Homogeneous Generalized Cylinders : Constraints from Contour

Ari David Gross

Vision/Robotics Lab
Columbia University

New York City, N.Y., 10027 ari@division.columbia.edu

Abstract

Generalized cylinders are a flexible, loosely defined class of parametric shapes capable of modeling many real-world objects. Straight homogeneous generalized cylinders are an important subclass of generalized cylinders whose cross sections are scaled versions of a reference curve. In this paper, the constraints imposed by the contour image of a straight homogeneous generalized cylinder on the underlying 3D object are studied. The main result of this paper is that there are *two parameters unconstrained by the contour image* such that, if these two parameters can be determined using some other method (e.g., intensity-based methods), the entire shape of the object (modulo scale) can be determined. After defining generalized cylinders and straight homogeneous generalized cylinders, the author shows that projective invariants of straight homogeneous generalized cylinders reduce the rotational transformations between object- and viewer- centered coordinate systems that need to be considered. This simplifies the ensuing analysis. Next, a proof is given to show that an infinite class of *contour-equivalent* straight homogeneous generalized cylinders exists that can be constructed from a *canonical* generalized cylinder by varying the slant parameter. It is then shown that this class can vary in quantitative Gaussian curvature at a surface point corresponding to a *fixed* image point. Next, a constructive proof is given that a second *contour-equivalent* class exists, this time generated by varying the translation parameter of the generalized cylinder axis with respect to the cross-section curve. This class is shown to have variations in both quantitative and qualitative Gaussian curvature. A method for recovering the *generalized ruling* of a straight homogeneous generalized cylinder contour is then described. Finally, it is shown that the two unconstrained parameters used in constructing the *contour-equivalent* classes are the *only* free parameters *not* constrained by image contour.

1. Introduction

A generalized cylinder (hereafter GC) is a solid defined by its axis, cross-section, and sweeping rule. Generalized cylinders were first proposed by Binford [1] as a class of parametric shapes that is very flexible and capable of modeling many different types of objects. GCs seem *general* enough to represent many real-world objects yet *sufficiently well-defined* that one is tempted to recover their shape from image data. They have been the topic of considerable research within the vision community [2]-[4],[6]-[9],[12],[13]-[18]. An important subclass of GCs is that of straight homogeneous generalized cylinders (hereafter SHGCs), where the axis is straight and cross-section curves are scaled versions of a reference curve (defined in section 2). The study of SHGC contour constraints is the subject of this paper.

SHGCs have proven difficult to recover from monocular intensity images. Brooks' ACRONYM system [4] was successful at recovering a very restricted subclass of GCs from contour images. The subclass considered by Brooks in the ACRONYM system consisted of GCs with a circular or simple polygonal cross section, straight or circular spine, and linear or bilinear sweeping rule. Even with this restricted subset of GCs, ACRONYM was only successful at recovering shape from image contour because it was matching to an *a priori* set of models. In fact, a monocular contour image of SHGCs (and certainly of GCs) is insufficient to yield a unique solution, as will be shown in this paper.

Recovery methods for SHGCs from monocular intensity images have tended to *fit* the image SHGC using only image contour information, as in [4],[9],[11],[20]. The *underconstrained* nature of the SHGC *shape from contour* problem has been compensated for by either considering restricted classes of SHGCs (e.g., surfaces of revolution), invoking heuristic methods, or having an *a priori* set of models to match to. In an effort to derive an *overconstrained* method that uses additional image information (e.g., intensity data), the author decided that a mathematical study of SHGC contour constraints was necessary to determine precisely what constraints an SHGC contour imposes on the underlying object. The results of that study, described in this paper, show that exactly two parameters remain unconstrained by SHGC contour. Those parameters are the *slant* and *translation* (with respect to the reference cross-section curve) of the SHGC axis. The

author has since shown [7] that, given certain assumptions regarding the SHGC and the imaging model, photometric invariants exist in the intensity image to recover these two parameters. The results of this contour constraint study can also be useful to heuristic-based methods in that the free parameters of SHGC contour have been clearly identified. In addition, by showing that two degrees of freedom exist that are *not* constrained by the contour image, it follows that no algorithm can claim to solve for the underlying shape of an SHGC strictly from contour information (e.g., [20]).

It is necessary to differentiate between terms referring to the 3D scene and terms referring to the image plane. A *contour generator* is defined as a 3D curve that generates the image contour. There are two kinds of contour generators, *limbs*, where the surface turns smoothly away from the viewer, and *edges*, where the surface orientation is discontinuous [14]. The 2D curves in the image corresponding to *limbs* and *edges* are referred to, respectively, as *image limbs* and *image edges*.

It is assumed in the sequel that projection from the scene onto the image plane is scaled orthographic. It is further assumed that both the sweeping rule function and the cross-section curve are twice continuously differentiable (C^2).

Curves on the SHGC surface of constant t are called *meridians*, while curves of constant z drawn on the SHGC surface are called *parallels* (see SHGC definition, section 2). This terminology is a generalization of that used for surfaces of revolution. The projections of meridians and parallels onto the image plane are referred to, in this paper, as *image meridians* and *image parallels*, respectively. The term *reference cross-section curve* (or simply *reference curve*) in the sequel refers to the 3D cross-section curve of the top SHGC plane. Its scaling is used as a *reference* to scale the other SHGC cross-sections.

Meridians and parallels, whose projections can be determined directly from the image contour (see section 6), provide a natural parameterization of an SHGC surface and seem to convey considerable information about the underlying shape, see figure 1 (as opposed to lines of curvature [3]). Nevertheless, in sections 4 and 5 of this paper it is shown that, without additional assumptions, *no algorithm can recover the shape of an SHGC from the contour image alone*. The underlying ambiguity is shown to have two parameters of freedom, axis slant and translation. The ambiguity is significant and can affect the sign and magnitude of Gaussian curvature at a point on the SHGC surface.

This paper avoids making certain assumptions that have been made by generalized cylinder researchers in the past and that are generally false. In particular, it can be shown that the contour generator of an SHGC is generally not planar, nor does it lie along a surface meridian, nor is it symmetric with respect to its axis (see [7],[8],[14],[9]).

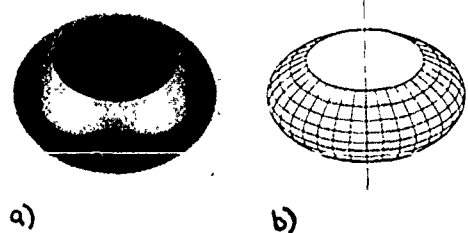


Figure 1. A straight homogeneous generalized cylinder a) the intensity image, b) the ruled contour.

The term *generalized ruling* in the sequel is an extension of the term *ruling* used for ruled surfaces. A *generalized ruling* on an SHGC surface (respectively an SHGC image) are its parallels and meridians (respectively image parallels and image meridians). An example of an SHGC intensity image and its *ruled* contour is given in figure 1.

In section 2, a definition of generalized cylinders and straight homogeneous generalized cylinders is given, as well as a discussion of coordinate systems and viewpoint assumptions. After defining SHGCs, useful properties of SHGCs are derived including expressions for the normal, limb equation, and Gaussian curvature (section 3). Next, a proof is given that a contour-equivalent class of SHGCs can be constructed by varying the axis slant parameter. This class, it is shown, exhibits quantitative Gaussian curvature variation (section 4). Then, it is shown that a contour-equivalent class of SHGCs can be constructed by varying the axis translation parameter, and that the class exhibits both quantitative and qualitative Gaussian curvature variation (section 5). In section 6, a method is derived for recovering the *generalized ruling* of an SHGC image given its contour. Finally, it is shown (section 7) that if, in addition to the SHGC contour, the two contour-equivalent parameters of axis slant and translation are known, then the shape of the underlying SHGC can be uniquely determined modulo scale.¹

2. Generalized Cylinders: Definitions and Assumptions

First, a definition for generalized cylinders as used in this paper.

Definition: A *generalized cylinder* is the solid swept by a planar cross-section as it is moved and deformed along an axis.

2.1 Straight Homogeneous Generalized Cylinders

A definition is now given for SHGCs as the term will be used in this paper.

Definition: An *SHGC* is a GC with the following properties: the axis is straight; the cross-section curve is a simple, smooth curve orthogonal to the axis; the cross-sections are deformed only by scaling; the scaling factor can be parameterized as a function of position along the axis;

SHGCs can be precisely defined with respect to the orthonormal coordinate system $(O, \vec{i}, \vec{j}, \vec{k})$, where O is a point on the axis, and (\vec{i}, \vec{j}) is a vector basis for the reference cross-section plane. An explicit parameterization for the swept surface of an SHGC, given as a function of z and t , can be written as

$$\vec{OP}(z, t) = r(z)p(t)\vec{i} + r(z)q(t)\vec{j} + z\vec{k} \quad (2.1)$$

where the function r is the sweeping rule of the SHGC and the functions p and q are, respectively, the components of the cross-section curve in the \vec{i} and \vec{j} directions. The sweeping rule r is presumed to be strictly positive (to avoid self-intersections), while p and q can be positive, zero, or negative. Note that this definition allows the cross-section curve to intersect the SHGC axis. Curves on the SHGC surface of constant t are called *meridians* while curves of constant z drawn on the SHGC surface are called *parallels*. Both the sweeping rule function and the cross-section curve are assumed to be twice continuously differentiable (C^2).²

The above definition for SHGCs is quite general. SHGCs as defined above are *much* more general than superquadrics, as they subsume the superquadric class even if the SHGC cross-section curve is restricted to being superelliptic. It also subsumes the class studied by Ponce et al [14], where the cross-section curve is restricted to being a piecewise polar function (assuming the knot points are C^2). The definition for SHGC in this paper, however, is a strict subset of the class defined by Ponce in [16], where the sweeping rule r is not required to be a function of z . This latter definition for SHGCs, though, is difficult to use as it includes instances of surfaces that are not regular (e.g., self-intersections), as noted in [16].

Note that the definition for SHGCs given here does not require the SHGC axis be contained within the closed cross-section curve (as in [14]). As a result, SHGCs with the axis external to the cross-section curve do not

always appear to have a straight axis. This class of SHGCs allows for a wide variety of shapes. Examples of these SHGCs are shown in figure 2. In particular, it should be observed that the banana-shaped object in figure 2.c can be parameterized as an SHGC with respect to *fixed* coordinate system $(O, \vec{i}, \vec{j}, \vec{k})$ rather than by a Frenet frame, as is typically done for curved objects, (see [15]).

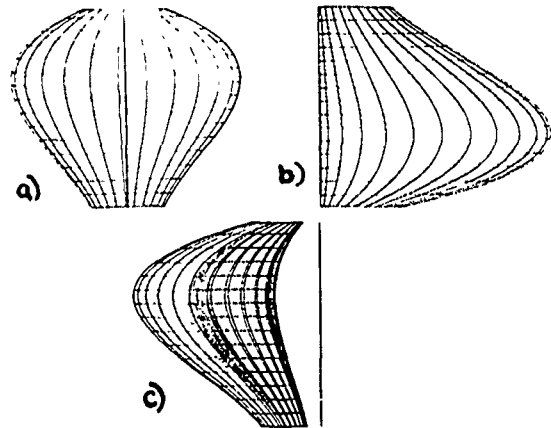


Figure 2. Three SHGCs: a. surface of revolution object; b. parabolic-sided object; c. banana-shaped object.

2.2 A Preliminary: The 2D Intersecting Tangents Lemma

As a preliminary to discussing viewpoint assumptions and coordinate systems, a previous result for SHGC projective contour invariants due to Ponce et al. [14] needs to be cited.

2D intersecting tangents lemma: For any two contour points with the same z value, the tangents to the contours intersect on the image axis.

This property of SHGC contour is helpful in the recovery process since it is a projective invariant of SHGC contour, i.e., it holds regardless of the direction from which the object is viewed. Ponce et al. [14] have demonstrated that this lemma can be used as the basis for recovering the image axis of an SHGC. It is interesting to note [14] that this lemma is valid in two dimensions but not in three, i.e., in general, the 3D tangents at points along the contour generator with the same z value do *not* intersect along the 3D SHGC axis.

Since the class of SHGCs considered in this paper is *more general* than that defined by Ponce, it is necessary to prove the lemma for the class of SHGCs defined in this paper. This is done in [7]. The 2D tangent lemma is illustrated in figure 3.

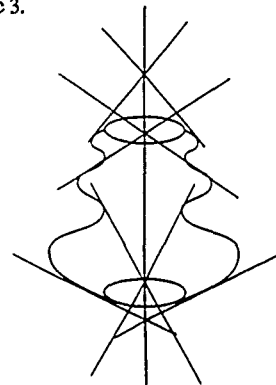


Figure 3. Example of the 2D tangent lemma for SHGC contour: tangents to contour points with the same z value intersect on the image of the axis.

2.3 The Viewpoint Assumption

In this section, the following assumption concerning viewpoint direction is motivated.

Slant Viewpoint Assumption: Without loss of generality, the only rotation

1. Recovering the shape of the object modulo scale is considered optimal since absolute scale cannot be recovered from a monocular intensity image under scaled orthographic projection.

2. The assumption that functions r , p , and q be C^2 is not strictly necessary. The results in this paper are valid even if the functions are only C^0 and piecewise C^2 . The C^2 assumption is made in this paper in order to simplify the contour analysis.

that needs to be considered in analyzing the constraints of SHGC contour on the shape of the underlying 3D object is that of slant towards (or away from) the viewer.

Consider an SHGC originally aligned with the viewer reference frame, where the viewer reference frame is given by the orthonormal coordinate system $(O, \mathcal{U}, \mathcal{V}, \mathcal{W})$, where \mathcal{V} is the viewer direction. The SHGC is parameterized in its own coordinate system, having an object-centered orthonormal basis $(\vec{i}, \vec{j}, \vec{k})$. The SHGC is originally in canonical position with respect to the viewer reference frame, i.e., the vectors \vec{i} and \vec{k} , \mathcal{V} and \vec{i} , and \mathcal{W} and \vec{j} are respectively parallel (see figure 4.a). Assume the SHGC \mathcal{OP} is then rotated in space; the rotated SHGC can be parameterized by

$$\mathcal{OP}' = R_\phi R_\psi R_\omega \mathcal{OP} \quad (2.2)$$

where ϕ , ψ , and ω are the Euler angles expressing the rotation about the \mathcal{V} , \mathcal{W} , and \mathcal{U} axes respectively, and R_ϕ , R_ψ , and R_ω are the corresponding rotation matrices. Clearly, after the initial rotation around the \mathcal{U} axis, the resulting SHGC can still be considered in canonical alignment with respect to the $(\mathcal{U}, \mathcal{V}, \mathcal{W})$ viewer-centered reference frame if the initial cross section functions p and q are replaced with new cross-section functions p' and q' (see figure 4.b).

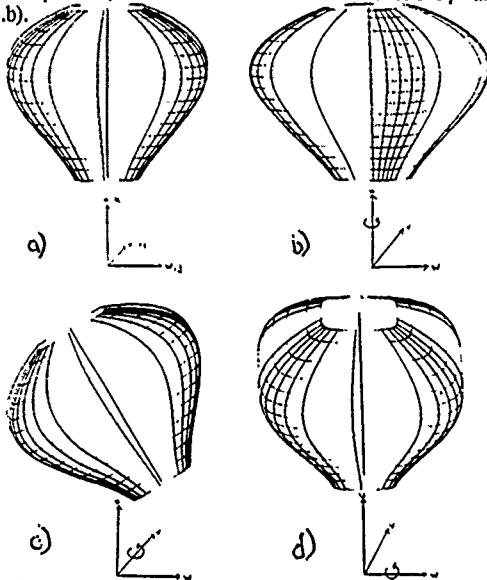


Figure 4. The coordinate system used to define a straight homogeneous generalized cylinder: a. in canonical position; b. rotated around the \mathcal{U} axis; c. rotated around the \mathcal{W} axis; d. rotated around the \mathcal{V} axis, i.e. slanted an angle of β towards the viewer.

The last rotation, by an angle of ϕ with respect to the \mathcal{V} axis, rotates the projected contour in the image plane but does not modify it in any other way (see figure 4.c). But from the 2D intersecting tangents lemma (section 2.2), an algorithm exists for recovering the SHGC image axis as in [14]. Thus this image plane rotation R_ϕ can be reversed by finding the image axis, which is a projection of the object-centered \vec{k} axis onto the image plane, and then undoing the R_ϕ rotation by bringing the image axis into alignment with (parallel to) the viewer-centered \mathcal{U} axis. Thus, without loss of generality, the only SHGC rotation (from canonical alignment) considered in the sequel is towards or away from the viewer, i.e., around the \mathcal{W} axis (see figure 4.d). This rotation around the \mathcal{W} axis is referred to as the SHGC slant. This justifies the slant viewpoint assumption stated above.

2.4 The Coordinate Systems

Suppose the viewing direction \mathcal{V} is given by its spherical coordinates (α, β) in $(O, \vec{i}, \vec{j}, \vec{k})$. Based on the preceding argument (section 2.3), without loss of generality, α can be set equal to zero. The resulting orthonormal basis of the viewer reference frame $(\mathcal{U}, \mathcal{V}, \mathcal{W})$ is defined by

$$\mathcal{U} = -\cos \beta \vec{i} + \sin \beta \vec{k}, \quad \mathcal{V} = \sin \beta \vec{i} + \cos \beta \vec{k}, \quad \mathcal{W} = \vec{j} \quad (2.3)$$

Consider the image of an SHGC for some viewing angle, $0 < \beta < \pi/2$, as shown in figure 4.d. Let $Q(z, t)$ be the projection of surface point $P(z, t)$ onto the image plane. It can be written as

$$\mathcal{OQ}(z, t) = r q \mathcal{W} + (z \sin \beta - r \cos \beta p) \mathcal{U} \quad (2.4)$$

The projection of a given cross-section curve $z = z_i$ is given by $\mathcal{OQ}(z_i, t)$. It is easily seen that each projected cross-section curve is a scaled version of a projected reference curve. This will prove useful for ruling the contour of the SHGC (section 6).

3. Properties of an SHGC

In this section, some properties of an SHGC are derived that will be used in the next several sections of the paper. First, surface properties of the SHGC are studied, then projective properties are considered.

3.1 Surface Properties

Consider a particular SHGC S_0 , whose swept out surface (as defined in equation (2.1)) can be written as

$$\mathcal{OP}_0(z, t) = r_0(z) p_0(t) \vec{i} + r_0(z) q_0(t) \vec{j} + z \vec{k}; \quad z_1 \leq z \leq z_2. \quad (3.1)$$

where $r_0(\cdot)$ is the sweeping rule of SHGC S_0 , and $p_0(\cdot)$ and $q_0(\cdot)$ are its respective cross-section functions along the \vec{i} and \vec{j} axes. Hereafter, function variables and variable subscripts are generally omitted.

For \mathcal{OP}_0 , the partial derivatives are given by

$$\frac{\partial \mathcal{OP}_0}{\partial z} = r' p \vec{i} + r' q \vec{j} + \vec{k} \quad (3.2)$$

$$\frac{\partial \mathcal{OP}_0}{\partial t} = r p' \vec{i} + r q' \vec{j} \quad (3.3)$$

The normal vector on the SHGC surface is obtained by taking the vector cross product of the partial derivative vectors. After taking the vector product and rescaling by a factor of $\frac{1}{r}$, the normal can be written as

$$\vec{N}_0 = q' \vec{i} - p' \vec{j} + r'(p' q - q' p) \vec{k} \quad (3.4)$$

To obtain an expression for the unit normal, equation (3.4) is divided its magnitude, whose squared value is given by

$$\|\vec{N}_0\|^2 = q'^2 + p'^2 + r'^2(p' q - q' p)^2 \quad (3.5)$$

Having derived expressions for the partial derivatives and surface normal, an expression can now be derived for the Gaussian curvature at a point on the surface. To compute the Gaussian curvature, second derivatives of S_0 with respect to z and t are required, for which one obtains

$$\begin{aligned} \frac{\partial^2}{\partial z^2} \mathcal{OP}_0 &= r'' \cdot (p \vec{i} + q \vec{j}) \\ \frac{\partial^2}{\partial z \partial t} \mathcal{OP}_0 &= r' \cdot (p' \vec{i} + q' \vec{j}) \\ \frac{\partial^2}{\partial t^2} \mathcal{OP}_0 &= r \cdot (p'' \vec{i} + q'' \vec{j}) \end{aligned} \quad (3.6)$$

The Gaussian curvature of the surface at a point $P_0(z, t)$ is given by (see [5])

$$K = \frac{e g - f^2}{E G - F^2}, \quad (3.7)$$

where the coefficients of the second and first fundamental forms in the basis $(\partial/\partial z, \partial/\partial t)$ are given by e, f, g , and E, F, G , respectively. These coefficients (see [5]), where \cdot is the vector dot product, are given by

$$e = \frac{1}{\|\vec{N}_0\|} \vec{N}_0 \cdot \frac{\partial^2}{\partial z^2} \mathcal{OP}_0 = \frac{1}{\|\vec{N}_0\|} r'' (p' q - q' p)$$

$$f = \frac{1}{\|\vec{N}_0\|} \vec{N}_0 \cdot \frac{\partial^2}{\partial z \partial t} \mathcal{OP}_0 = 0$$

$$g = \frac{1}{\|\vec{N}_0\|} \vec{N}_0 \cdot \frac{\partial^2}{\partial t^2} \mathcal{OP}_0 = \frac{1}{\|\vec{N}_0\|} r (q' p'' - p' q'')$$

$$E = \frac{\partial}{\partial z} \bar{O} \bar{P}_0 \cdot \frac{\partial}{\partial z} \bar{O} \bar{P}_0 = r'^2 (p^2 + q^2) + 1 \quad (3.8)$$

$$F = \frac{\partial}{\partial z} \bar{O} \bar{P}_0 \cdot \frac{\partial}{\partial t} \bar{O} \bar{P}_0 = r r' (p' + q' q')$$

$$G = \frac{\partial}{\partial t} \bar{O} \bar{P}_0 \cdot \frac{\partial}{\partial t} \bar{O} \bar{P}_0 = r^2 (p'^2 + q'^2)$$

Substituting into equation (3.7) the expressions for e , f , g , and E , F , G from equation (3.8), we obtain, after some simplification, an expression for Gaussian curvature K of SHGC S_0 given by

$$K(z, t) = \frac{r'' \cdot (p' q' - q p') \cdot (q' p'' - p' q'')}{r [p'^2 + q'^2 + r'^2 (p' q' - q p')^2]} \quad (3.9)$$

The above expression for K will be compared in sections 4 and 5 with the Gaussian curvature of SHGCs having the same contour as that of SHGC S_0 .

3.2 Projective Properties

In this section, projective properties of SHGC S_0 are derived. Since we are interested in the projective properties of an SHGC, some viewing direction is required. Based on the *slant viewpoint assumption* (section 2.3), it is sufficient to consider a viewer reference frame that has only a slant rotation with respect to the object-centered coordinate system. The resulting orthonormal basis of the viewer reference frame $(\bar{u}, \bar{v}, \bar{w})$ is defined in equation (2.3), and is given by

$$\bar{u} = -\cos \beta \bar{i} + \sin \beta \bar{k}, \quad \bar{v} = \sin \beta \bar{i} + \cos \beta \bar{k}, \quad \bar{w} = \bar{j}$$

If $Q_0(z, t)$ is the projection of surface point $P_0(z, t)$ onto the image plane, it can be written as

$$\bar{Q}_0(z, t) = r_0 q_0 \bar{w} + (z \sin \beta - r_0 p_0 \cos \beta) \bar{u} \quad (3.10)$$

The *limbs* of an SHGC are the loci of surface points satisfying the condition $\bar{N} \cdot \bar{v} = 0$, where \bar{N} is the surface normal vector. A *limb equation* is an equation that is satisfied by exactly those points lying on the limbs of the SHGC surface.

The normal \bar{N}_0 given in equation (3.4), when converted into the $(\bar{u}, \bar{v}, \bar{w})$ basis defined above, has the form

$$\bar{N}_0(z, t) = [-\cos \beta q' + \sin \beta r' (p' q - q' p)] \bar{u} + [\sin \beta q' + \cos \beta r' (p' q - q' p)] \bar{v} + -p' \bar{w} \quad (3.11)$$

Since the limb points of SHGC S_0 are exactly those points on the surface that satisfy $\bar{N}_0 \cdot \bar{v} = 0$, the limb equation for SHGC S_0 when viewed from viewing direction \bar{v} is given by

$$\sin \beta q' + \cos \beta r' (p' q - q' p) = 0 \quad (3.12)$$

Having analyzed some of the projective properties of SHGC S_0 , one can describe the contour of S_0 . The 3D *contour generator* of S_0 is comprised of the SHGC's edges and limbs, as described in section 1. Analogously, let C_0 , the *image contour* of SHGC S_0 , be comprised of *image limbs* L_0 and *image edges* E_0 . The contour of S_0 can then be written as

$$C_0 = L_0 \cup E_0 \quad (3.13)$$

where L_0 is given by equation (3.10) restricted to the set of (z, t) satisfying limb equation (3.12), and E_0 is also given by equation (3.10) where z is restricted to $z = z_1$ or $z = z_2$ (as in equation (3.1)).

These projective properties of SHGC S_0 will be used in the next two sections to show that classes of SHGCs can be constructed such that their contours are equivalent to that of SHGC S_0 .

4. Slant Contour-Equivalent Classes of SHGCs

In this section, a class of SHGCs is constructed and analyzed. The class is constructed from SHGC S_0 defined in section 3. It is constructed by changing the slant of the SHGC object with respect to the viewer. The slant of the object depends on the viewing direction \bar{v} . The basic method used in the construction of *slant contour-equivalent* SHGC S_i is to vary the viewing direction \bar{v}_i with respect to the SHGC while modifying its other parameters

so that its contour C_i is identical to C_0 , the contour of SHGC S_0 . Demonstrating that such an infinite class of SHGCs can be constructed from canonical SHGC S_0 is proof of the fact that the slant of an SHGC cannot be solved for strictly from contour.

Section 4.1 defines the class of SHGCs $\{S_i\}$ that are *slant contour-equivalent* to SHGC S_0 .³ Section 4.2 analyzes the projective properties of the class. Section 4.3 proves that this class is in fact contour-equivalent to S_0 , i.e., that $C_i = C_0$. Section 4.4 examines the Gaussian curvature properties of the class.

4.1 Definition of a Slant Contour-Equivalent Class

Let SHGC S_i be defined by

$$\bar{O} \bar{P}_i(s_i(z), t) = f_i(s_i(z)) \cdot c_i \cdot p_0(t) \bar{i} + f_i(s_i(z)) q_0(t) \bar{j} + s_i(z) \bar{k}; \quad (4.1)$$

$$s_i(z_1) \leq s_i(z) \leq s_i(z_2)$$

where $c_i \geq 1$ is the *stretching factor* associated with S_i , $\gamma_i = \cos^{-1}(\cos \beta / c_i)$ is the slant of SHGC S_i in the viewer direction, $s_i(z) = z \frac{\sin \beta}{\sin \gamma_i}$ is the adjusted z value along the SHGC axis, and $f_i(s_i(z)) = r_0(z)$ is the modified sweeping rule. The functions p_0 , q_0 and r_0 are as defined for SHGC S_0 in equation (3.1). The c_i term in equation (4.1) can be thought of as the factor by which the cross-section curve for S_i has been stretched in the \bar{i} direction, while γ_i is the slant of SHGC S_i with respect to the viewer. In the sequel, function variables and variable subscripts are generally omitted.

It is clear from examining equation (4.1) that stretching factor c_i is the only free parameter.⁴ Once c_i is fixed, SHGC S_i is well-defined. The viewing direction for S_i , given by slant angle γ_i , is a function of the stretching factor c_i .

As in section 3, suppose \bar{v}_i , the viewing direction for SHGC S_i , is given by its spherical coordinates (α_i, γ_i) in $(O, \bar{i}, \bar{j}, \bar{k})$. Based on the *slant viewpoint assumption* (section 2.3), the viewing direction \bar{v}_i is restricted to a slant towards or away from the viewer, i.e., $\alpha_i = 0$. The resulting orthonormal basis of the viewer reference frame $(\bar{u}_i, \bar{v}_i, \bar{w}_i)$ is defined by

$$\bar{u}_i = -\cos \gamma_i \bar{i} + \sin \gamma_i \bar{k}, \quad \bar{v}_i = \sin \gamma_i \bar{i} + \cos \gamma_i \bar{k}, \quad \bar{w}_i = \bar{j} \quad (4.2)$$

where \bar{v}_i is the viewing direction, γ_i the slant angle for SHGC S_i , and (\bar{w}_i, \bar{u}_i) is an orthonormal basis for the image plane.

The partial derivatives for $\bar{O} \bar{P}_i$ are given by

$$\frac{\partial \bar{O} \bar{P}_i}{\partial z} = f_i s_i c_i p \bar{i} + f_i s_i q \bar{j} + s_i \bar{k} \quad (4.3)$$

$$\frac{\partial \bar{O} \bar{P}_i}{\partial t} = f_i c_i p' \bar{i} + f_i q' \bar{j} \quad (4.4)$$

Taking the vector product of the partial derivatives and rescaling the result by a factor of $\frac{1}{f_i s_i}$, we have an expression for the normal vector of SHGC S_i given by

$$\bar{N}_i = q' \bar{i} - p' \bar{j} + f_i c_i (p' q - q' p) \bar{k} \quad (4.5)$$

4.2 Slant Contour-Equivalent SHGCs: Projective Properties

The contour C_i is generated by projecting SHGC S_i onto the image plane along viewing direction \bar{v}_i . Let $Q_i(s_i(z), t)$ be the projection of the point $P_i(s_i(z), t)$ into the image plane. We have

$$\bar{Q}_i(s_i(z), t) = f_i q \bar{w} + (\sin \gamma_i \cdot s - \cos \gamma_i \cdot f_i c_i p) \bar{u} \quad (4.6)$$

where $s = s_i(\cdot)$, $f = f_i(\cdot)$, and $c = c_i$, as defined in equation (4.1). Substituting for $s_i(z)$ and c_i , we have

$$\bar{Q}_i(s_i(z), t) = f_i q \bar{w} + (\sin \beta z - \cos \beta f_i p) \bar{u}$$

By definition (from equation (4.2)), $f_i(s_i(z)) = r(z)$. So this equation further reduces to

³ Contour-equivalence depends on viewer direction. In section 3, the viewer direction was given as \bar{v} , which was equivalent to a slant of β in the viewer direction.

⁴ β is the slant of SHGC S_0 with respect to the viewer reference frame. It is fixed with respect to the class of SHGCs defined in equation (4.1).

$$O\tilde{Q}_i(s, t) = r q \tilde{w} + (z \sin \beta - r \cos \beta p) \tilde{u} \quad (4.7)$$

But since this equation has the same form as equation (3.10), it follows that

$$O\tilde{Q}_i(s_i(z), t) = O\tilde{Q}_0(z, t) \quad (4.8)$$

Consider a point $P_0(z_1, t_1)$ on the surface of S_0 and a point $P_i(s(z_1), t_1)$ on the surface of S_i . It follows from equation (4.8) that these two points map onto the same point in the image plane.

The normal \tilde{N}_i in equation (4.5) above, when converted into the viewer reference frame given in equation (4.2) (where \tilde{v}_i is the viewing direction), is given by

$$\tilde{N}_i(s, t) = \begin{bmatrix} -\cos \gamma q' + \sin \gamma f, c(p'q - q'p) \\ \sin \gamma q' + \cos \gamma f, c(p'q - q'p) \end{bmatrix} \tilde{u} + \begin{bmatrix} \sin \gamma q' + \cos \gamma f, c(p'q - q'p) \\ -\cos \gamma q' + \sin \gamma f, c(p'q - q'p) \end{bmatrix} \tilde{v} - c p' \tilde{w} \quad (4.9)$$

Limb points on SHGC S_i are exactly those points on the surface that have $\tilde{N}_i \cdot \tilde{v}_i = 0$. So the limb equation for S_i is given by

$$\sin \gamma q' + \cos \gamma f, c(p'q - q'p) = 0 \quad (4.10)$$

4.3 A Proof of Slant Contour-Equivalence

Image contours are the projections of two kinds of contour generators, limbs, where the surface turns smoothly away from the viewer, and edges, where surfaces nonsmoothly intersect. For the case of SHGC S_i defined in equation (4.1), the contour C_i can be written as

$$C_i = L_i \cup E_i \quad (4.11)$$

where L_i is given by equation (4.7) restricted to the set of (s, t) satisfying limb equation (4.10), and E_i is also given by equation (4.7) where z is restricted to $s(z) = s(z_1)$ or $s(z) = s(z_2)$ (as in equation (4.1)).

To prove that C_0 (the contour of SHGC S_0) defined in equation (3.13), and C_i (the contour of SHGC S_i) defined in equation (4.11), are equivalent, we first prove the correctness of two related lemmas.

Slant Lemma 1: Let SHGC S_i , having stretching factor c_i , be constructed from basis SHGC S_0 as defined in equation (4.1). Let L_0 be the image limb of S_0 when viewed from direction \tilde{v}_0 , where \tilde{v}_0 is given in equation (2.3). Let L_i be the image limb of S_i when viewed from direction \tilde{v}_i , where \tilde{v}_i is given in equation (4.2). Then the image limbs for S_i and S_0 are identical, that is

$$L_i = L_0 \quad (4.12)$$

Proof: To show that $L_i = L_0$, it is sufficient, using equation (4.8), to show that a point $P_0(z, t)$ on the surface of S_0 is a limb point iff a corresponding point $P_i(s(z), t)$ on the surface of S_i is a limb point.

Since by definition from equation (4.1), we have

$$f_i(s_i(z)) = r(z)$$

Differentiating this equation with respect to z yields

$$f, s_i = r' \quad (4.13)$$

Differentiating $s(z)$, as defined in equation (4.1), with respect to z , one obtains

$$s_i = \frac{\sin \beta}{\sin \gamma} \quad (4.14)$$

Substituting for s_i , equation (4.13) can be rewritten as

$$f_i = r' \frac{\sin \gamma}{\sin \beta} \quad (4.15)$$

Substituting this expression for f_i into the limb equation (4.10) derived for SHGC S_i and multiplying by $\frac{\sin \beta}{\sin \gamma}$, the limb equation for S_i can be rewritten as

$$\sin \beta q' + \cos \gamma r' c(p'q - q'p) = 0 \quad (4.16)$$

From the definition of γ_i in equation (4.1), one can write

$$\cos \gamma_i = \frac{\cos \beta}{c_i} \quad (4.17)$$

Substituting for $\cos \gamma$ in equation (4.16) yields

$$\sin \beta q' + \cos \beta r'(p'q - q'p) = 0 \quad (4.18)$$

But this equation is identical to the limb equation for SHGC S_0 , given in equation (3.12). Thus, a surface point $P_0(z, t)$ on SHGC S_0 is a limb point iff a surface point $P_i(s(z), t)$ on SHGC S_i is a limb point. This is sufficient, using equation (4.8), to show that $L_i = L_0$, so the lemma is proved.

There is another lemma, this one concerning image edges, that needs to be proved in order to prove contour equivalence.

Slant Lemma 2: Let SHGC S_i , having stretching factor c_i , be constructed from basis SHGC S_0 as defined in equation (4.1). Let E_0 be the image edges of S_0 when the viewing direction is \tilde{v}_0 , given in equation (2.3). Let E_i be the image edges of S_i when the viewing direction is \tilde{v}_i , given in equation (4.2). Then the image edges for S_i and S_0 are identical, that is

$$E_i = E_0 \quad (4.19)$$

Proof: Edges are defined as curves where the surface orientation is discontinuous. Since the cross-section curve and sweeping rule, as defined in this paper, are both twice continuously differentiable (C^2), the SHGC surface is smooth and edges occur only at the endpoints (i.e., top and bottom cross-section planes).

The 3D edges for SHGC S_0 are generated by keeping z fixed, $z = z_1$ or $z = z_2$, and letting t vary. An image edge, as defined in section 1, is not simply the projection of the corresponding 3D edge on the SHGC surface since part of the 3D edge may be self-occluded. Without loss of generality, consider the edge resulting from the intersection of the bottom cross-section plane, $z = z_2$, with the swept surface of SHGC S_0 . The image projection of this 3D curve is given by

$$O\tilde{Q}_0(z_2, t) = r(z_2)q\tilde{w} + [z_2 \sin \beta - r(z_2) \cos \beta p] \tilde{u}; \quad t_a \leq t \leq t_b$$

If there is no self-occlusion (i.e., the entire projected 3D edge curve is visible in the image), then $O\tilde{Q}_i(s(z_2), t_a) = O\tilde{Q}_i(s(z_2), t_b)$.

From the definition of SHGC S_i (equation (4.1)) and the previous lemma, it follows that the image projection of the corresponding 3D curve on SHGC S_i is given by

$$O\tilde{Q}_i(s(z_2), t) = f(s(z_2))q\tilde{w} + [\sin \gamma_i s(z_2) - \cos \gamma_i f(s(z_2))c p] \tilde{u}; \quad t_a \leq t \leq t_b$$

The fact that the intervals on t are identical follows from the fact that $P_i(s(z), t)$, satisfies its limb equation exactly when $P_0(z, t)$ satisfies its respective limb equation. But since, from equation (4.8), we have that

$$O\tilde{P}_i(s(z), t) = O\tilde{P}_0(z, t),$$

it follows that $E_i = E_0$ and the lemma is proved.

A theorem of contour-equivalence can now be proved.

Slant Axis Theorem: Let SHGC S_i , having stretching factor c_i , be constructed from basis SHGC S_0 as defined in equation (4.1). Let C_i be the image contour of S_i when the viewing direction is \tilde{v}_i , given in equation (4.2). Then the image contours for S_i and S_0 are identical, that is

$$C_i = C_0 \quad (4.20)$$

Proof: The theorem follows immediately from slant lemmas 1 and 2, since

$$C_i = L_i \cup E_i = L_0 \cup E_0 = C_0$$

4.4 Slant Contour-Equivalent SHGCs: Gaussian Curvature

In section 4.2 it was established that $P_0(z, t)$ and $P_i(s(z), t)$ map onto the same point in the image. Furthermore, in the previous section it was shown that the class of SHGCs defined in section 4.1 have contours equivalent to that of SHGC S_0 . In this section, a fixed image point is studied to derive a form for the Gaussian curvature of the corresponding point on the surface when that surface is allowed to vary among members of the contour-equivalent class defined in equation (4.1).

Consider the expression for the normal \vec{N}_i of SHGC S_i given in equation (4.5). Substituting into this equation the expression for f , given by equation (4.15), the normal equation can be rewritten as

$$\vec{N}_i = q' \vec{i} - c p' \vec{j} + r' \frac{\sin \gamma}{\sin \beta} c (p' q - q' p) \vec{k} \quad (4.21)$$

whose squared magnitude is given by

$$|\vec{N}_i|^2 = q'^2 + c^2 p'^2 + r'^2 \left[\frac{\sin \gamma}{\sin \beta} \right]^2 c^2 (p' q - q' p)^2 \vec{k}^2 \quad (4.22)$$

Since the sweeping rule f and the cross-section curves p and q are assumed to be C^2 , the second derivatives with respect to z and t are given by

$$\frac{\partial^2}{\partial z^2} \vec{O}P_i = f_{,zz} \cdot s_i^2 \cdot (c p' \vec{i} + q' \vec{j}) \quad (4.23)$$

$$\frac{\partial^2}{\partial z \partial t} \vec{O}P_i = f_{,zt} \cdot s_i \cdot (c p' \vec{i} + q' \vec{j}) \quad (4.24)$$

$$\frac{\partial^2}{\partial t^2} \vec{O}P_i = f_{,tt} \cdot c p'' \vec{i} + f_{,tt} q'' \vec{j} \quad (4.25)$$

The Gaussian curvature of the surface at a point $\vec{O}P_i(s, t)$ is given by equation (3.7), where e, f, g , and E, F, G are, respectively, coefficients of the second and first fundamental forms of the surface in the basis $(\frac{\partial}{\partial z} \vec{O}P_i, \frac{\partial}{\partial t} \vec{O}P_i)$. These coefficients (see [5]), where \cdot is the vector dot product, are given by

$$e = \frac{1}{|\vec{N}_i|} \vec{N}_i \cdot \frac{\partial^2}{\partial z^2} \vec{O}P_i = \frac{1}{|\vec{N}_i|} f_{,zz} s_i^2 c (p' q' - q' p')$$

$$f = \frac{1}{|\vec{N}_i|} \vec{N}_i \cdot \frac{\partial^2}{\partial z \partial t} \vec{O}P_i = 0$$

$$g = \frac{1}{|\vec{N}_i|} \vec{N}_i \cdot \frac{\partial^2}{\partial t^2} \vec{O}P_i = \frac{1}{|\vec{N}_i|} f_{,tt} c (q' p'' - p' q'') \quad (4.26)$$

$$E = \frac{\partial}{\partial z} \vec{O}P_i \cdot \frac{\partial}{\partial z} \vec{O}P_i = s_i^2 [f_{,z}^2 (c^2 p'^2 + q'^2) + 1]$$

$$F = \frac{\partial}{\partial z} \vec{O}P_i \cdot \frac{\partial}{\partial t} \vec{O}P_i = f_{,zt} s_i (c^2 p' p' + q' q')$$

$$G = \frac{\partial}{\partial t} \vec{O}P_i \cdot \frac{\partial}{\partial t} \vec{O}P_i = f_{,t}^2 (c^2 p'^2 + q'^2)$$

An expression for s_i is given in equation (4.14). Using equation (4.13), one obtains an equation for $f_{,zz}$ given by

$$f_{,zz} = \frac{r_{,zz}}{s_i^2} \quad (4.27)$$

After substituting these expressions for s_i and $f_{,zz}$ into the equations for the first and second fundamental forms, and after some algebraic manipulation, a solution is obtained for the Gaussian curvature K of SHGC S_i at point $(s, (z), t)$ on the surface given by

$$K_i(s, t) = \frac{a_i^2 b_i^2 r'' \cdot (p' q' - q' p') \cdot (q' p'' - p' q'')}{r [a_i^2 b_i^2 p'^2 + b_i^2 q'^2 + a_i^2 r'^2 (p' q' - q' p')^2]^2} \quad (4.28)$$

where $a_i = c_i = \frac{\cos \beta}{\cos \gamma}$ and $b_i = \frac{\sin \beta}{\sin \gamma}$. Terms involving the sweep rule f ,

have been replaced with equivalent terms using the sweep rule r for SHGC S_i . This allows for comparison of expressions for Gaussian curvature given by the above equation and equation (3.9).

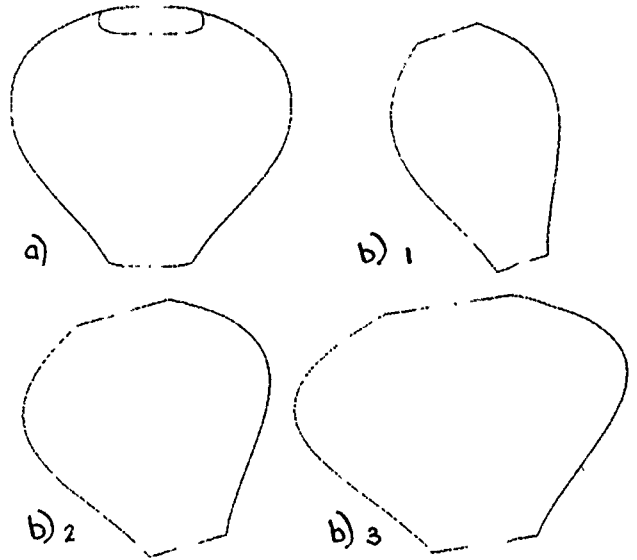


Figure 5. Example of slant contour-equivalent SHGCs: a. identical slant contour-equivalent view for 3 SHGCs; b. side view of same 3 SHGCs seen with slant decreasing in the viewer direction.

It can be seen from this equation that as stretching factor c , changes so does the Gaussian curvature. In fact, the Gaussian curvature at a point on this surface goes to zero as $k \rightarrow \infty$. The Gaussian curvature will tend towards ∞ at a point where $r' = 0$, $p' = 0$, and $k \rightarrow \infty$. It can also be determined from equation (4.28) that the sign of the Gaussian curvature will not vary at a given surface point for any member of the *slant contour-equivalent* class of SHGCs, i.e., the sign of the Gaussian curvature at $P_i(s(z), t)$ is the same as the sign of the Gaussian curvature at $P_0(z, t)$. Thus, Gaussian curvature variations among this contour-equivalent class are only quantitative not qualitative.

Some examples of slant contour-equivalent SHGCs are shown in figure 5. In Figure 5a, a contour-equivalent view is shown of an SHGC. This image could have been generated from an infinite family of slant contour-equivalent SHGCs. For three such contour-equivalent SHGCs, a side view (rotating the SHGC 30 degrees around its axis) is given in figure 5b. It can be seen that though they are contour-equivalent from a given viewing direction, they are in fact very different shapes. The reader can also verify that the Gaussian curvature at corresponding surface points varies only quantitatively.

5. Translation Contour-Equivalent Classes of SHGCs

In this section, a class of SHGCs is constructed and analyzed. The class is constructed from canonical SHGC S_0 defined in section 3. It is constructed by changing the translation of the SHGC object with respect to the reference cross-section curve. For this class of SHGCs, the viewing direction ν is the same as for SHGC S_0 (used to generate contour C_0). The basic method used in the construction of *translation contour-equivalent* SHGC S_i is to vary the translation parameter of the axis with respect to the reference cross-section curve, while modifying its other parameters so that its contour C_i is identical to C_0 (contour of SHGC S_0). *Demonstrating that such an infinite class of SHGCs can be constructed from canonical SHGC S_0 is proof of the fact that the axis translation of an SHGC cannot be solved for strictly from contour.*

Section 5.1 defines the class of SHGCs $\{S_i\}$ that are *translation contour-equivalent* to SHGC S_0 . Section 5.2 analyzes the projective properties of the class. Section 5.3 proves that this class is in fact contour-equivalent to S_0 , i.e., that $C_i = C_0$. Section 5.4 examines the Gaussian curvature properties of the class.

5.1 Definition of a Translation Contour-Equivalent Class

Let SHGC S_i be defined by

$$\vec{OP}_i(s_i(z), t) = \left[f_i(s_i(z)) \cdot (p(t) + h_i) - h_i \right] \vec{i} + f_i(s_i(z)) \cdot q(t) \vec{j} + s_i(z) \vec{k}; \quad (5.1)$$

$$s_i(z_1) \leq s_i(z) \leq s_i(z_2).$$

where h_i is the translation factor associated with SHGC S_i , $s_i(z) = z - \frac{h_i(1-r(z))}{\tan \beta}$ is the position of the cross-section curve along the SHGC axis, $f_i(s_i(z)) = r(z)$ is the modified sweeping rule, where $r(z)$ is the sweeping rule defined for SHGC S_0 in equation (3.1). The $p(\cdot)$ and $q(\cdot)$ terms in the equation are identical to those for SHGC S_0 , i.e., S_i has the same cross-section curve as S_0 . The h_i term in equation (5.1) can be thought of as the factor by which the reference cross-section curve for SHGC S_i has been translated from its axis in the \vec{i} direction.

It is clear from examining equation (5.1) that axis translation factor h_i is the only free parameter.⁵ Once h_i is fixed, SHGC S_i is well-defined. The viewing direction for S_i is given by \vec{v} just as it is for S_0 , i.e., it is not a function of SHGC S_i . The orthonormal basis of the viewer reference frame ($\vec{u}, \vec{v}, \vec{w}$) is exactly as defined in equation (2.3).

The partial derivatives for \vec{OP}_i with respect to z and t are given by

$$\frac{\partial \vec{OP}_i}{\partial z} = f_i \cdot s_i \cdot (p+h) \vec{i} + f_i \cdot s_i \cdot q \vec{j} + s_i \vec{k} \quad (5.2)$$

$$\frac{\partial \vec{OP}_i}{\partial t} = f_i p' \vec{i} + f_i q' \vec{j} \quad (5.3)$$

Taking their vector product, we have

$$\vec{N}_i = q' \vec{i} - p' \vec{j} + f_i \cdot (p'q - q'(p+h)) \vec{k} \quad (5.4)$$

5.2 Translation Contour-Equivalent SHGCs: Projective Properties

The contour C_i is generated by projecting SHGC S_i onto the image plane along viewing direction \vec{v} . Let $Q_i(s_i(z), t)$ be the projection of the point $P_i(s_i(z), t)$ into the image plane. We have

$$\vec{OQ}_i(s(z), t) = f_i q \vec{w} + \left[\sin \beta s(z) - \cos \beta \cdot (f_i \cdot (p+h) - h_i) \right] \vec{u} \quad (5.5)$$

where $s = s(\cdot)$, $f = f_i(\cdot)$, and $c = c_i$, as defined in equation (5.1).

Substituting for $s(z)$ in equation (5.5), using the definition for $s(z)$ in equation (5.1), and simplifying, yields

$$\vec{OQ}_i(s, t) = f_i q \vec{w} + \left[z \sin \beta - \cos \beta \cdot h_i \cdot (1-r(z)) - \cos \beta \cdot (f_i \cdot (p+h) - h_i) \right] \vec{u} \quad (5.6)$$

But by definition (equation (5.1)), $f_i(s_i(z)) = r(z)$. Substituting for f_i , the above can be written as

$$\vec{OQ}_i(s, t) = r q \vec{w} + (z \sin \beta - r \cos \beta p) \vec{u} \quad (5.7)$$

which is identical to the form for \vec{OP}_0 derived in equation (3.10). Consequently, it follows that

$$\vec{OP}_i(s_i(z), t) = \vec{OP}_0(z, t) \quad (5.8)$$

Thus, a point $P_0(z, t)$ on the surface of SHGC S_0 and a point $P_i(s(z), t)$ on the surface of SHGC S_i project onto the same point in the image plane.

The normal \vec{N}_i in equation (5.4) above, when converted into the viewer reference frame given in equation (2.3) (where \vec{v} is the viewing direction), is given by

$$\vec{N}(\alpha, z, t) = \left[-\cos \beta q' + \sin \beta f_i (p'q - q'(p+h)) \right] \vec{u} + \left[\sin \beta q' + \cos \beta f_i (p'q - q'(p+h)) \right] \vec{v} + -p' \vec{w} \quad (5.9)$$

Limb points on SHGC S_i are exactly those points on the surface that have $\vec{N}_i \cdot \vec{v} = 0$. So the limb equation for S_i is given by

⁵ As in the previous section, β is the slant of SHGC S_0 with respect to the viewer reference frame. It is fixed with respect to the class of SHGCs defined in equation (5.1).

$$\sin \beta q' + \cos \beta f_i (p'q - q'(p+h)) = 0 \quad (5.10)$$

5.3 A Proof of Translation Contour-Equivalence

To prove that contour C_i of SHGC S_i , defined in equation (5.1), is equivalent to contour C_0 of canonical SHGC S_0 , two lemmas need to be established, (analogous to the method used for proving contour equivalence in section 4).

Translation Lemma 1: Let SHGC S_i , defined in equation (5.1) and having translation factor h_i , be constructed from basis SHGC S_0 (defined in equation (3.1)). Let L_i be the image limb of S_i , where the viewing direction is \vec{v} , given by equation (2.3). Then the image limbs for S_i and S_0 are identical, that is

$$L_i = L_0 \quad (5.11)$$

Proof: To show that $L_i = L_0$, it is sufficient, using equation (5.8), to show that a point $P_0(z, t)$ on the surface of S_0 is a limb point iff a point $P_i(s(z), t)$ on the surface of S_i is a limb point.

By definition (equation (5.1)), one can write

$$f_i(s_i(z)) = r(z)$$

Differentiating this equation with respect to z yields

$$f_i s_i = r' \quad (5.12)$$

Taking the derivative of $s(z)$, defined in equation (5.1), with respect to z , one obtains

$$s_i = 1 + \frac{h_i r'}{\tan \beta} \quad (5.13)$$

Substituting for s_i , equation (5.12) can be rewritten as

$$f_i = \frac{r' \tan \beta}{\tan \beta + h_i r'} \quad (5.14)$$

Substituting this expression for f_i into limb equation (5.10) and rescaling by a factor of $\frac{\tan \beta + h_i r'}{\tan \beta}$, one obtains

$$\cos \beta q' \cdot (\tan \beta + h_i r') + \cos \beta r' \cdot (p'q - q'(p+h)) = 0 \quad (5.15)$$

which simplifies to

$$\sin \beta q' + \cos \beta r' \cdot (p'q - q'p) = 0 \quad (5.16)$$

But this equation is identical to limb equation (3.12) derived for SHGC S_0 . Thus, a surface point $P_0(z, t)$ on SHGC S_0 is a limb point iff a surface point $P_i(s(z), t)$ on SHGC S_i is a limb point. This is sufficient, in conjunction with equation (5.8), to show that $L_i = L_0$, which proves the lemma.

There is another lemma, this one concerning image edges, that needs to be proved.

Translation Lemma 2: Let E_i be the image edges of S_i , defined in equation (5.1), when the viewing direction is \vec{v} (given in equation (2.3)). Then the image edges for S_i and S_0 are identical, that is

$$E_i = E_0 \quad (5.17)$$

Proof: Similar to proof of lemma 2, section 4.3.

From the two preceding lemmas, the following theorem immediately follows.

Translated Axis Theorem: Let SHGC S_i , having translation factor h_i , be constructed from basis SHGC S_0 as defined in equation (5.1). Let C_i be the image contour of S_i when the viewing direction is \vec{v} (given in equation (2.3)). Then the image contours for S_i and S_0 are identical, that is

$$C_i = C_0 \quad (5.18)$$

5.4 Translation Contour-Equivalent SHGCs: Gaussian curvature

In this section, a point in the image is studied to derive a form for the Gaussian curvature of the corresponding point on the surface when that surface is allowed to vary among members of the contour-equivalent class defined in equation (5.1).

Consider the expression for the normal \vec{N}_i of SHGC S_i given in equation (5.4). Substituting into this equation the expression for f_i given by equation (5.14), the normal equation can be rewritten as

$$\vec{N}_i = q' \vec{i} - p' \vec{j} + \left[\frac{r' \tan \beta}{\tan \beta + h r'} \right] (p' q - q' (p + h)) \vec{k} \quad (5.19)$$

For computing the first and second fundamental forms, the unit normal is required. So the expression for the normal \vec{N}_i needs to be divided by its magnitude, whose squared value is given by

$$\|\vec{N}_i\|^2 = q'^2 - p'^2 + \left[\frac{r' \tan \beta}{\tan \beta + h r'} \right]^2 (p' q - q' (p + h))^2 \quad (5.20)$$

The second derivatives with respect to z and t are given by

$$\frac{\partial^2}{\partial z^2} \vec{O}P_i = (f_{ii} \cdot s_i^2 + f_i \cdot s_{ii}) \cdot ((p + h) \vec{i} + q \vec{j}) + s_{ii} \vec{k} \quad (5.21)$$

$$\frac{\partial^2}{\partial z \partial t} \vec{O}P_i = f_i \cdot s_i \cdot p' \vec{i} + f_i \cdot s_i \cdot q' \vec{j} \quad (5.22)$$

$$\frac{\partial^2}{\partial t^2} \vec{O}P_i = f_i \cdot p'' \vec{i} + f_i \cdot q'' \vec{j} \quad (5.23)$$

The Gaussian curvature of the surface at a point $\vec{O}P_i(s, t)$ is given by (see [5])

$$K = \frac{e g - f^2}{E G - F^2} \quad (5.24)$$

where the coefficients of the second and first fundamental forms in the basis $((\partial/\partial z) \vec{O}P_i, (\partial/\partial t) \vec{O}P_i)$, are given by e, f, g , and E, F, G , respectively. These coefficients, where the \cdot operator is the vector dot product, are given by

$$e = \frac{1}{\|\vec{N}_i\|} \vec{N}_i \cdot \frac{\partial^2}{\partial z^2} \vec{O}P_i = \frac{1}{\|\vec{N}_i\|} f_{ii} s_i^2 ((p + h) q' - q p')$$

$$f = \frac{1}{\|\vec{N}_i\|} \vec{N}_i \cdot \frac{\partial^2}{\partial z \partial t} \vec{O}P_i = \frac{1}{\|\vec{N}_i\|} 0 = 0$$

$$g = \frac{1}{\|\vec{N}_i\|} \vec{N}_i \cdot \frac{\partial^2}{\partial t^2} \vec{O}P_i = \frac{1}{\|\vec{N}_i\|} f_i (q' p'' - p' q'') \quad (5.25)$$

$$E = \frac{\partial}{\partial z} \vec{O}P_i \cdot \frac{\partial}{\partial z} \vec{O}P_i = s_i^2 f_i^2 ((p + h)^2 + q^2) + s_i^2$$

$$F = \frac{\partial}{\partial z} \vec{O}P_i \cdot \frac{\partial}{\partial t} \vec{O}P_i = f_i f_i (p' (p + h) + q' q)$$

$$G = \frac{\partial}{\partial t} \vec{O}P_i \cdot \frac{\partial}{\partial t} \vec{O}P_i = f_i^2 (p'^2 + q'^2)$$

From the definition of $s_i(z)$ in equation (5.1), it follows that

$$s_i = 1 + \frac{h r'}{\tan \beta}$$

Using equation (5.14), an equation for f_{ii} can be written as

$$f_{ii} = r_{ii} \left[\frac{\tan \beta}{\tan \beta + h r_i} \right]^3 \quad (5.26)$$

After substituting the expressions for s_i and f_{ii} derived above into the coefficients for the fundamental forms given in equation (5.25), and after some algebraic manipulation, we obtain an expression for the Gaussian curvature K given by

$$K_i(s, t) = \frac{d^3 r'' ((p + h) q' - q p') (q' p'' - p' q'')}{r (p'^2 + q'^2 + d^2 r'^2 ((p + h) q' - q p')^2)} \quad (5.27)$$

where $d = \frac{\tan \beta}{\tan \beta + h r'}$ and h is the translation of the reference cross-section curve from the axis for SHGC S_i . Terms involving the sweep rule f_i have been replaced with equivalent terms using the sweep rule r for SHGC S_0 . This allows for comparison of expression for Gaussian curvature given by the above equation and equation (3.9).

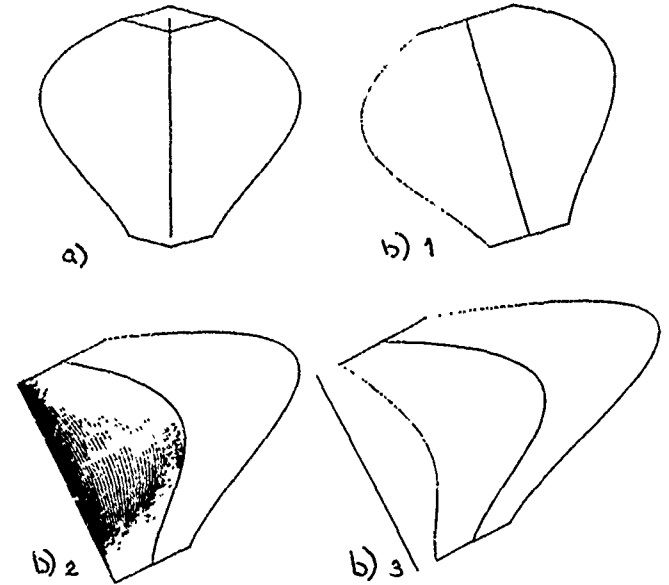


Figure 6. Example of translation contour-equivalent SHGCs: a. identical translation contour-equivalent view for 3 SHGCs; b. side view for same 3 SHGCs seen with different axis translation values.

It can be readily seen from this equation that as the translation factor h changes so does the Gaussian curvature. In fact, the Gaussian curvature at a point on the surface goes to zero as $d \rightarrow \infty$; this occurs when $\tan \beta = -h r'$. Also, it can be seen that $K \rightarrow 0$ as $h \rightarrow \infty$. Because of the d^3 term in the numerator of the equation, it is clear that by varying h the Gaussian curvature can be made to change not only quantitatively but also qualitatively (positive to negative or the reverse).

From the expression for Gaussian curvature K in equation (5.27) and from the above analysis, it is clear that the Gaussian curvature at a surface point P , corresponding to fixed image point I , can vary both quantitatively and qualitatively.

Along the contour generator, there is an additional constraint, i.e., limb equation (5.10) is satisfied. The expression for the limb equation given by equation (5.15), after rescaling by $\frac{1}{\cos \beta}$, can be rewritten as

$$\frac{q'}{r'} (\tan \beta + h r') = (p + h) q' - q p' \quad (5.28)$$

Replacing $(p + h) q' - q p'$ in equation (5.27) with $\frac{q'}{r'} (\tan \beta + h r')$, one gets a simplified expression for K given by

$$K_i(s, t) = \frac{d^2 r'' \tan \beta q' (q' p'' - p' q'')}{r r' (p'^2 + q'^2 (1 + \tan^2 \beta))^2} \quad (5.29)$$

where $d = \frac{\tan \beta}{\tan \beta + h r'}$ and h is the translation of the reference cross-section curve from the axis for SHGC S_i . Since the only term involving h in this expression is the d term in the numerator, and it is squared, it follows that the sign of the Gaussian curvature along an SHGC contour generator does not vary among members of a translation contour-equivalent class. This is consistent with a result by Koenderink [10] that the qualitative nature of the surface (e.g., elliptic vs. hyperbolic) at a point on the contour generator can be determined from its projected contour.

Examples of translation contour-equivalent SHGCs are shown in figure 6. In Figure 6a, a translation contour-equivalent view is shown of the

contour of an SHGC. This image could have been generated from an infinite family of translation contour-equivalent SHGCs. For three such contour-equivalent SHGCs, a side view (rotating the SHGC 90 degrees around its axis) is shown in figure 6b. As in the previous section, it can be seen that though these SHGCs are contour-equivalent from a certain viewing direction, they are in fact very different shapes. The reader can also verify that the Gaussian curvature at corresponding surface points can vary both quantitatively and qualitatively.

6. Ruling Over Generalized Cylinders

In this section, a method is described for finding image parallels and meridians from the contour image (image limbs and image edges) of an SHGC. This method is referred to as recovering the *generalized ruling* of the SHGC contour, or simply as *ruling* the contour (see section 1).

In general, most non-occluded image cross-section curves tend to have two or more points of intersection with the image limbs, which suggests a method for ruling the SHGC image. This method naturally assumes the image axis has already been recovered [7],[14]. Algorithms using the SHGC contour to recover the image axis of an SHGC, where the cross-section function is assumed polar with respect to the axis, are given in [14] (though the robustness of such algorithms is not assured). In [7] the 2D axis lemma, on which the algorithm for recovering the image axis is based, is generalized to SHGCs with arbitrary, simple C^2 cross-sections, as defined in equation (3.1).

Assuming the image axis has been recovered, the reference curve can be *rescaled* with respect to the axis so that it touches the bounding contour at 2 or more points, without any point on this rescaled curve extending beyond the contour. Using a non-accidentalness alignment criterion, it is assumed that if such a scaling exists, it indicates that the image parallel at this point along the axis has been correctly recovered. This method allows an image parallel to be drawn at any desired point on the image axis. Connecting corresponding points of image parallels together using interpolating splines provides an approximation to the image meridians.

This technique is illustrated in figure 7 which shows an SHGC contour and its image axis. Also shown in the figure are scaled versions of the image cross-section with respect to a certain point along the image axis. It can be seen that only at one such scaling does the image cross-section curve exactly touch the contour in two places; at these points, the image cross-sectional tangent is parallel to the contour tangent. At every other scaling, the cross-section curve is either contained entirely within or extends beyond the bounding image contour. Thus, this *osculating* image parallel is taken to be the correct scaling of the image parallel at this point along the image axis.

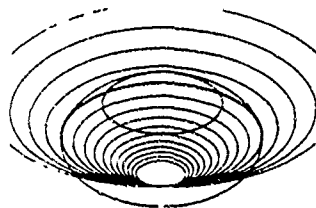


Figure 7. A method for ruling the SHGC contour: different scalings of the image parallel and the correct scaling.

The method described above works when the SHGC axis is contained within the cross-section curve. In a case where the axis is not contained within the cross-section curve,⁶ a more general method is required, described in [7], that involves both translating and rescaling the image cross-section curve.

In [16], two SHGCs are said to be equivalent if they can be deduced from each other through any sequence of a certain set of transformations. One of these transformations involves *scaling the sweeping rule by a non-zero constant while scaling the reference cross-section by the inverse of*

that constant. Using that transformation, one can decide on a particular parameterization from among this equivalence class by, somewhat arbitrarily, setting the scaling function of the top cross section curve to one, i.e., $r(z_1) = 1$. Having done so, it is clear from equation (2.4) that the scaling factor is also known for all the image parallels detected using the method described above.

7. Unconstrained Parameters of SHGC Contour

Consider a contour whose image axis has been recovered (section 2.3) and that has been ruled (section 6). The scaling of an image parallel can be determined, but it is not clear *where* this image parallel is located on the image axis. This allows for a translation parameter not constrained by contour, as discussed in section 5. In addition, the image parallel is a projection of a 3D parallel curve. The actual slant of the 3D parallel curve, however, cannot be determined from the contour (ruled or otherwise), as demonstrated in section 4. From sections 4 and 5, then, it has been established that *at least two parameters are unconstrained by SHGC contour*. In this section the author will show that, given a ruled SHGC contour and its image axis, there are *exactly two* parameters unconstrained by SHGC contour.

Consider the SHGC whose contour is shown in figure 8a. The underlying SHGC object can be described by a reference cross-section curve and axis origin (where the SHGC axis intersects the cross-section plane), as shown in figure 8b., and by a sweeping rule (given with respect to the reference cross-section curve), as shown in figure 8c.⁷

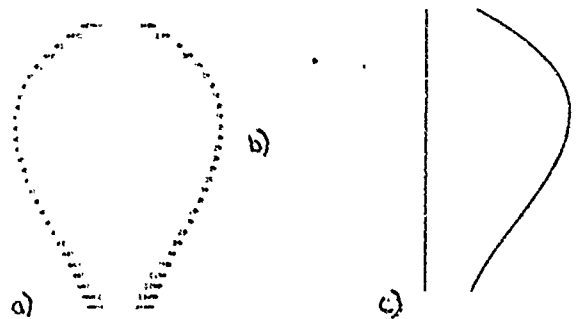


Figure 8. Description of an SHGC : a. SHGC contour; b. cross-section curve and axis origin; c. sweeping rule

To show that exactly two parameters, axis slant and translation, are unconstrained by SHGC contour, it suffices to give an algorithm to completely describe the shape of the underlying SHGC (modulo scale) if these two parameters were known. If axis slant angle β were known, the 3D cross-section curve can certainly be determined from its projected image cross-section curve, given in equation (2.4).⁸ Let a point on the SHGC image axis be selected as the origin of the cross section curve. Then, knowing the axis slant, the p and q values at any point along the cross-section curve can be solved for with respect to the cross-section origin. If the translation parameter h were known, defined in equation (5.1) with respect to the origin of the cross-section curve, then the origin of the SHGC axis would be known, i.e., the point at which the SHGC axis intersects the reference cross-section curve.

A ruled SHGC contour contains information regarding the sweeping rule of the underlying SHGC. The problem is that without knowing the SHGC origin, the sweeping rule cannot be solved for. To see this, consider the contour shown in figure 9a. Figure 9b shows the cross-section curve, axis origin and sweeping rule for an SHGC that is consistent with the contour of figure 9a. In figure 9c, the cross-section curve, origin, and sweeping rule of another such SHGC is shown, this SHGC is also consistent with the SHGC contour of figure 9a. The cross-section curve, axis origin and sweeping rule of a third such SHGC (consistent with figure 9a) is shown in figure 9d. This illustrates the fact that a sweeping rule can only be determined from an SHGC contour once the origin of the SHGC's coordinate system is known.

⁷ The object description is, of course, only modulo scale since *absolute* scale cannot be determined from a monocular intensity image.

⁸ A *general viewpoint assumption* is implicitly assumed here, i.e., 2 points on the 3D cross section curve do not project onto the same point in the image.

⁶ The definition of an SHGC given in this paper does not assume the axis is contained within the cross-section curve.

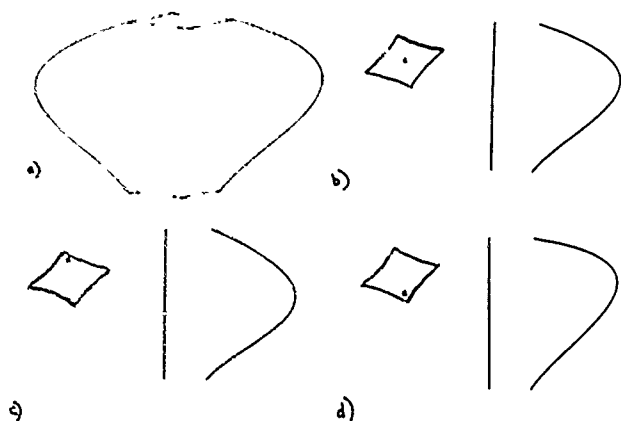


Figure 9. Different descriptions of the same SHGC contour: a. an SHGC contour; b. cross-section curve, axis origin, and sweep rule of SHGC 1; c. cross-section curve, axis origin, and sweep rule of SHGC 2; d. cross-section curve, axis origin, and sweep rule of SHGC 3;

Once the axis translation parameter h is known, however, the sweeping rule can be determined from the SHGC contour in a straightforward manner. Consider again the contour of figure 9.a. Assume that the axis slant and translation have been recovered using a method relying on additional image information (e.g., heuristics, intensity data). Since we have the slant value, the 3D cross-section curve can be determined, as mentioned above. In addition, the axis translation is known; this is equivalent to knowing where the axis intersects the cross-section plane of the reference curve. If one has an initial contour image and then is able to derive the axis slant and translation, all that then remains to completely describe the underlying SHGC (modulo scale) is the sweeping rule. So for example, having the image shown in figure 9.a and the cross-section curve and SHGC origin shown in figure 9.b, one needs a way to compute the sweeping rule of the SHGC in figure 9.b. But this can easily be accomplished by ruling the axis as densely as necessary to interpolate the sweeping rule curve. Knowing the axis translation, the method for ruling the contour described in section 6 can be used to obtain equi-spaced image parallels. Having found these equi-spaced parallels, one can easily approximate the SHGC sweeping rule by spline-based interpolative methods.

Thus, if the two unconstrained parameters of axis slant and translation are known, the 3D cross-section curve, axis origin, and the sweeping rule can be determined from the ruled contour image. This motivates the following theorem:

Contour Constraint Theorem: *There are exactly 2 parameters not constrained by SHGC contour, i.e., axis slant and translation. If the values of these two parameters are known, the underlying SHGC object can be completely described (modulo scale).*

8. Conclusion

This paper described results from a mathematical study of the constraints of SHGC contour on the underlying 3D SHGC object. The results from this study have provided the impetus for the author to derive an algorithm relying on photometric invariants [8] of the SHGC intensity image to

recover those parameters unconstrained by SHGC image contour.

In this paper, two classes of contour-equivalent SHGC are defined and their surface and projective properties analyzed. One contour-equivalent class is constructed by varying the axis slant, while the other class is constructed by varying the axis translation. Construction of these contour-equivalent classes is proof of the fact that these two parameters cannot be solved for strictly from the contour image. Among the surface properties studied for these contour-equivalent classes was that of Gaussian curvature. It is shown that slant contour-equivalent SHGCs can vary quantitatively but not qualitatively, while translation contour-equivalent SHGCs can also have qualitative Gaussian curvature variation.

A method of ruling the contour image, determining image parallels and meridians, was then described. It was shown that, if the axis slant and translation can be derived using other methods not based solely on contour, the entire shape of the underlying SHGC can be determined (modulo scale).

The methodology of this paper was to thoroughly analyze the constraints of SHGC contour. Knowing these constraints, intensity-based methods have subsequently been found to recover the parameters unconstrained by contour. In a sequel to this paper, the author describes such intensity-based methods and their implementation [8].

References

- [1] Binford, T. O., Visual Perception by computer, *Proceedings IEEE Conf. on Systems and Control*, Miami, December, 1971.
- [2] Brady, J.M., and Yuille, A., An extremum principle for shape from contour, MIT, AI Lab., MIT-AIM 711, 1983.
- [3] Brady, J.M., Ponce, J., Yuille, A., Asada, H., Describing Surfaces, in *Proceedings of the 2nd Int'l Symposium on Robotics Research*, Harasuja and Inoue (ed.), MIT Press, 1985.
- [4] Brooks, R. A., Symbolic reasoning among 3-D models and 2-D images, *Artificial Intelligence* 17, 285-348, 1981.
- [5] do Carmo, M.P., *Differential geometry of curves and surfaces*, Prentice-Hall, Inc., 1976.
- [6] Fearing, F.S., Tactile sensing, perception, and shape interpretation, PhD thesis, Dept. of Electrical Engineering, Stanford University, Dec. 1987.
- [7] Gross, A.D., *Recovery of straight homogeneous generalized cylinders from a single intensity view*, CS Technical Report CUCS-494-89, Columbia University, 1989.
- [8] Gross, A.D., Boulton, T.E., Recovery of generalized cylinders from a single intensity view, submitted to *Proceedings of Darpa IU Workshop*, 1990.
- [9] Horrad, R. and Brady, J.M., On the geometric interpretation of image contours, in *Proc. First Int'l Conference on Computer Vision*, London, U.K., June 1987.
- [10] Koenderink, J.J., What does occluding contour tell us about solid shape?, *Perception*, vol. 13, pages. 321-330, 1984.
- [11] Marr, D., Analysis of occluding contour, *Proc. of Royal Society of London B-197*, pp. 441-475, 1977.
- [12] Marr, D., *Vision*, San Francisco, CA, WH Freeman, 1982.
- [13] Nalwa, Vic, Line-drawing interpretation: Bilateral symmetry, in *Proceedings of the Image Understanding Workshop*, vol. 2, Feb., 1987, pp. 956-967.
- [14] Ponce, J., Chelberg, D., and Mann, W., Invariant properties of straight homogeneous Generalized Cylinders and their Contours, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. II, No. 9, September 1989.
- [15] Ponce, J., Chelberg, D., and Mann, W., Analytical properties of Generalized Cylinders and their projections, in *Proceedings of the First International Conference on Computer Vision*, 1987.
- [16] Ponce, J., Straight Homogeneous Generalized Cylinders: Differential Geometry and Uniqueness Results, in *Proceedings of Computer Vision and Pattern Recognition Conference*, 327-334, June 1988.
- [17] Rao, K., and Medioni, G., Useful geometric properties of the generalized cone, in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, June 5-9, 1988, pp. 276-281.
- [18] Richetin, M., Dhome, M., and LaPrete, J.T., Inverse perspective transform from zero-curvature curve points application to the localization of some generalized cylinders, in *Proceedings of the 1989 Computer Vision and Pattern Recognition Conference*, San Diego, California, June 4-8, 1989.
- [19] Shafer, S., *Shadows and silhouettes in computer vision*, Kluwer Acad Publishers, Boston, 1985.
- [20] Ulupinar, F., and Nevatia, R., Using Symmetries for Analysis of Shape from Contour, in *Proceedings of the Second International Computer Vision Conference*, 414-426, Dec. 5-8, 1988.

SELF-OPTIMIZING CONTROL SYSTEM FOR ADAPTIVE IMAGE SEGMENTATION

Bir Bhanu, Sungkee Lee, and John Ming

Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

ABSTRACT

One of the fundamental weaknesses of current computer vision systems to be used in practical outdoor applications is their inability to adapt the segmentation process as real-world changes occur in the image. We present the first closed loop self-optimizing control system for adaptive image segmentation which incorporates a genetic optimization algorithm to adapt the segmentation process to changes in image characteristics caused by variable environmental conditions such as time of day, time of year, clouds, etc. The genetic algorithm efficiently searches the hyperspace of segmentation parameter combinations to determine the parameter set which maximizes the segmentation quality criteria. The goals of our adaptive image segmentation system are to provide continuous adaptation to normal environmental variations, to exhibit learning capabilities, and to provide robust performance when interacting with a dynamic environment. We present experimental results when segmentation quality is either a scalar or vector valued function and optimization technique is either a pure genetic algorithm or a combination of genetic algorithm and hill climbing. These results demonstrate the ability to adapt the segmentation performance automatically in outdoor color imagery.

1. INTRODUCTION

Image segmentation is typically the first, and most difficult task of any automated image understanding process. All subsequent interpretation tasks including object detection, feature extraction, object recognition, and classification rely heavily on the quality of the segmentation process. Despite the large number of segmentation techniques presently available,^{4,6} no general methods have been found that perform adequately across a diverse set of imagery. Only after many modifications to its control parameter set can any current segmentation technique be used to process the diversity of images found in real world applications.

When presented with a new image, selecting the appropriate set of algorithm parameters is the key to effectively segmenting the image. The image segmentation problem can be characterized by several factors which make the parameter selection process very difficult. *First*, most segmentation techniques contain numerous control parameters which must be adjusted to obtain optimal performance. The size of the parameter search space in these systems can be prohibitively large, unless it is traversed in a highly efficient manner. *Second*, the parameters within most segmentation algorithms typically interact in a complex, non-linear fashion, which makes it difficult or impossible to model the parameters' behavior in an algorithmic or rule-based fashion. *Third*, since variations between images cause changes in the segmentation results, the objective function that represents seg-

mentation quality varies from image to image. The search technique used to optimize the objective function must be able to adapt to these variations. *Finally*, the definition of the objective function itself can be subject to debate because there are no universally accepted measures of image segmentation quality.

Hence, we must apply a technique that can efficiently search the complex space of parameter combinations and locate the values which yield optimal results. The approach should not be dependent on the particular application domain nor should it have to rely on detailed knowledge pertinent to the selected segmentation algorithm. The key elements of our adaptive image segmentation system are: (1) A closed-loop feedback control technique that consists of a genetic learning component, an image segmentation component, and a segmented image evaluation component; (2) A genetic learning system that optimizes segmentation performance of each image and accumulates segmentation experience over time to reduce the effort needed to optimize the segmentation quality of succeeding images; (3) Image characteristics and external image variables are represented and manipulated using both numeric and symbolic forms within the genetic knowledge structure, only the segmentation parameters are represented and manipulated in binary strings; (4) Image segmentation performance is evaluated using five measures of segmentation quality that measure *global* characteristics of the entire image as well as *local* features of individual object regions; (5) The adaptive segmentation system is not dependent on any specific segmentation algorithm or type of sensor.

The focus of our work is not to develop yet another specialized segmentation algorithm that works only in a very limited domain on a few images, but is directed towards adapting the performance of a well known existing segmentation algorithm^{9,11} across a wide variety of environmental conditions that cause changes in the image characteristics. To date, no segmentation algorithm has been developed which can automatically generate an "ideal" segmentation result in one pass (or in an open loop manner) over a range of scenarios encountered in practical outdoor applications. Any technique, no matter how "sophisticated" it may be, will eventually yield poor performance if it can not adapt to the variations in outdoor scenes. Therefore, in this paper we attempt to address this fundamental bottleneck in developing "useful" computer vision systems for practical scenarios by developing a closed-loop system that automatically adapts the segmentation algorithm's performance by changing its control parameters and will be valid across a wide diversity of image characteristics and application scenarios. It should be noted that the performance of the adaptive algorithm will be limited by the capabilities of the segmentation algorithm, but the results will be optimal for a given image based on our

evaluation criteria.

2. SEGMENTATION AS A SEARCH PROBLEM

Figure 1 shows an outdoor image and the typical segmentation quality surface (discussed in Section 3.4) associated with the image in which only two segmentation parameters^{8,11} are being varied. Because of the large number of potential parameter combinations and the subtle interaction of the algorithm parameters, the objective function is complex, multimodal, and presents problems for many commonly used search and optimization techniques. The drawbacks to some of these methodologies for the segmentation optimization problem have been discussed in the paper by Bhanu, Lee and Ming.¹

Genetic algorithms^{2,3,5,7} which are designed to efficiently locate an approximate global maximum in a search space show great promise in solving the parameter selection problem encountered in the image segmentation task. Since they use simple recombinations of existing high quality individuals and a method of measuring current performance, they do not require complex surface descriptions, domain specific knowledge, or measures of goal distance. Moreover, due to the generality of the genetic process, they are independent of the segmentation technique used, requiring only a measure of performance (which we refer to as segmentation quality) for any given parameter combination.

Genetic algorithms can be briefly characterized by three main concepts: a Darwinian notion of fitness or strength which determines an individual's likelihood of affecting future generations through reproduction; a reproduction operation which produces new individuals by combining selected members of the existing population; and genetic operators which create new offspring based on the structure of their parents. A genetic algorithm maintains a constant-sized *population* of candidate solutions, known as individuals. The initial seed population from which the genetic process begins can be chosen randomly or on the basis of heuristics. At each iteration, known as a *generation*, each individual is evaluated and recombined with others on the basis of its overall quality or *fitness*. The expected number of times an individual is selected for recombination is proportional to its fitness relative to the rest of the population. New individuals are created using two main genetic recombination operators known as *crossover* and *mutation*. Crossover operates by selecting a random location in the genetic string of the parents (crossover point) and concatenating the initial segment of one parent with the final segment of the second parent to create a new child. A second child is simultaneously generated using the remaining segments of the two parents. Mutation provides for occasional disturbances in the crossover operation by inverting one or more genetic elements during reproduction. This operation insures diversity in the genetic strings over long periods of time and prevents stagnation in the convergence of the optimization technique.

3. ADAPTIVE IMAGE SEGMENTATION

Adaptive image segmentation requires the ability to modify control parameters in order to respond to changes that occur in the image as a result of varying environmental conditions. The block diagram of our approach is shown in Figure 2. After acquiring an input image, the system analyzes the image characteristics and passes this information, in conjunction with the observed external variables, to the genetic learning component. Using this data, the genetic learning system selects an appropriate parameter combination, which is passed to the image segmentation process. After the image has been segmented, the results are evaluated and an

appropriate reward is generated and passed back to the genetic algorithm. This process continues until a segmentation result of acceptable quality is produced. The details of each component in this procedure will be described in the following subsections.

3.1 Image Characteristics

We compute twelve first order image properties for each color component (red, green, and blue) of the image. These features include mean, variance, skewness, kurtosis, energy, entropy, x intensity centroid, y intensity centroid, maximum peak height, maximum peak location, interval set score, and interval set size. Since we use a black/white version of the image to compute edge information and object contrast during the evaluation process, we also compute the twelve features for the Y (luminance component) image as well. Combining the image characteristic data from these four components yields a list of 48 elements. In addition, we utilize two external variables, time of day and weather conditions. The external variables are represented symbolically in the list structure (e.g., time = 9am, 10am, etc. and weather conditions = sunny, cloudy, hazy, etc.). The distances between these values are computed symbolically when measuring image similarity. The two external variables are added to the list to create an image characteristic list of 50 elements.

3.2 Genetic Learning System

Fig. 3 shows a simple example of our genetic learning system. The image characteristics for a new image are compared with the individuals in the global population to obtain the initial seed for the local population. The global population represents the accumulated segmentation experience for all images that the system has processed whereas the local population contains the set of segmentation parameters processed by the genetic algorithm during the optimization of the current image. To obtain the initial local population (seed population) for a new image from the global population, a normalized Euclidean feature distance is computed from the new image to every member of the global population and this distance is used along with the fitness of each individual in the global population for selecting the closest individuals. Although we have limited the seed population to 3 in this example, our experiments utilize a seed population of 10 individuals. The global population holds 100 knowledge structures in order to maintain a diverse collection of segmentation experience. The parameter sets in the seed population are used to segment the image and the results are evaluated to generate a fitness for each individual. The fitness value (leftmost value in the list) varies from 0.0 to 1.0 and measures the quality of the segmentation parameter set. Note that only the fitness value and the action portion (segmentation parameters) of the knowledge structure are subject to genetic adaptation; the conditions (image characteristics) remain fixed for the life of the structure. If the fitness values are not acceptable, the individuals are recombined and the process repeats. Each pass through the loop (segmentation-evaluation-recombination), is known as a generation. The cycle continues until the maximum fitness achieved at the end of a generation exceeds some threshold. The global population is updated using the high quality members of the local population from the current image and the system is then ready to process another image.

3.3 Segmentation Algorithm

Since we are working with color imagery in our experiments, we have selected the well known Phoenix segmentation algorithm developed at Carnegie Mellon Univer-

sity.^{8,9,11} Phoenix⁸ contains fourteen different control parameters which are used to control the thresholds and termination conditions used within the algorithm. There are 10^{33} conceivable parameter combinations using these fourteen values. Of the fourteen values, we have selected two of the most critical parameters that affect the overall results of the segmentation process, *maxmin* and *hsmooth*. *Maxmin* specifies the lowest acceptable peak-to-valley-height ratio used when deciding whether or not to split a large region into two or more smaller parts. *Hsmooth* controls the width of the window used to smooth the histogram of each image region during segmentation. The use of only two parameters for the initial tests aids in the *visualization* of the optimization process since we can plot the associated segmentation quality corresponding to each parameter combination using a 3D plotting technique. Future research will incorporate a larger number of modifiable parameters.

3.4 Segmentation Evaluation

There are a large number of segmentation quality measures that have been suggested, although none have achieved widespread acceptance as a universal measure of segmentation quality. In order to overcome the drawbacks of using only a single quality measure, we have incorporated an evaluation technique that uses five different quality measures to determine the overall fitness for a particular parameter set. Most of the measures of segmentation performance that we have selected for this work have been proposed in the computer vision literature and similar measures have been recommended by DoDs Automatic Target Recognition Working Group (ATRWG) as good indicators of segmentation quality. The five segmentation quality measures are:

(1) *Edge-Border Coincidence*: Measures the overlap of the region borders in the image acquired from the segmentation algorithm relative to an edge image.

$$\text{Edge-border Coincidence} = \frac{n(E \cap S)}{n(E)}$$

where $n(A)$ = the number of elements in set A , $E \cap S = \{(x_k, y_k), k = 1, \dots, m \text{ where } (x_k, y_k) \in E \text{ and } S\}$, E is the set of pixels extracted by the edge operator, and S is the set of pixels found on the region boundaries of the segmented image.

(2) *Boundary Consistency*: Similar to edge-border coincidence, except that region borders which do not exactly overlap edges can be matched with each other. Also, region borders which do not match with any edges are used to penalize the segmentation quality.

$$\text{Boundary Consistency} = M, \text{ if } M \geq 0$$

$$= 0, \text{ if } M < 0$$

where

$$M = \frac{\left[\sum_i W_1 * (d_{\max} - d_i) \right]}{n(E)} - \frac{\left[W_2 * \sum \text{remaining pixels in } E \text{ and } S \right]}{n(E)},$$

$W_1 = 0.1$, $W_2 = 0.5$, $d_{\max} = 10$, and d_i = the distance to the nearest pixel.

(3) *Pixel Classification*: This measure is based on the number of object pixels classified as background pixels and the number of background pixels classified as object pixels.

$$\text{Pixel Classification} = N, \text{ if } N \geq 0$$

$$= 0, \text{ if } N < 0$$

where

$$N = 1 - \left[\frac{(n(A) - n(A \cap B)) + (n(B) - n(A \cap B))}{n(A)} \right],$$

$A \cap B = \{(x_k, y_k), k = 1, \dots, m \text{ where } (x_k, y_k) \in A \text{ and } B\}$, A is the set of object pixels in the groundtruth image, and B is the set of object pixels in the segmented image.

(4) *Object Overlap*: Measures the area of intersection between the object region in the ground truth image and the segmented image.

$$\text{Object Overlap} = \frac{n(A \cap B)}{n(A)}$$

where $A \cap B = \{(x_k, y_k), k = 1, \dots, m \text{ where } (x_k, y_k) \in A \text{ and } B\}$.

(5) *Object Contrast*: Measures the contrast between the object and the background in the segmented image, relative to the object contrast in the ground truth image.

$$\text{Object Contrast} = \frac{C_{SI}}{C_{GT}}, \text{ if } C_{GT} \geq C_{SI}$$

$$= \frac{C_{GT}}{C_{SI}}, \text{ if } C_{GT} < C_{SI}.$$

$$\text{where } C_{GT} = \left| \frac{I_A - I_X}{I_A} \right|, \quad C_{SI} = \left| \frac{I_B - I_Y}{I_B} \right|,$$

C_{GT} is the contrast of the object in the groundtruth image, C_{SI} is the contrast of the object in the segmented image,

I_R = average intensity of region $R = (1/R_{\max}) \sum_{j=1}^{R_{\max}} I(j)$, A is the set of object pixels in the groundtruth image, B is the set of object pixels in the segmented image, X is the set of pixels surrounding region A in the groundtruth image, and Y is the set of pixels surrounding region B in the segmented image.

The maximum and minimum values for each of the five segmentation quality measures are 1.0 and 0.0, respectively. The first two quality measures are *global* measures since they evaluate the segmentation quality of the whole image with respect to edge information. Conversely, the last three quality measures are *local* measures since they only evaluate the segmentation quality for the object regions of interest in the image. When an object is broken up into smaller parts during the segmentation process, only the largest region which overlaps the actual object in the image is used in computing the local quality measures. The three local measures require the availability of object ground truth information in order to correctly evaluate segmentation quality. Since we desire

good object regions as well as high quality overall segmentation results, we have combined global and local quality measures (with equal weighting) to obtain a *combined* segmentation quality measure that maximizes overall performance of the system. Figure 4 shows the surfaces defined for the five individual quality measures that are used to create the combined quality measure surface shown in Figure 1.

4. EXPERIMENTAL RESULTS

An initial database of outdoor imagery was collected to demonstrate the system's ability to adapt to real world conditions and produce the best segmentation result based on our evaluation criteria. The database consists of twenty frames that were collected approximately every 15 minutes over a 5 hour period (1:30 pm to 6:30 pm) using a JVC GXF700U color video camera. A representative subset of these images, shown in Figure 5, will be used to describe the complete experimental results. This type of image data simulates a photointerpretation scenario in which the camera position is fixed and the image undergoes significant change over time. Weather conditions in our image database varied from bright sun to overcast skies. Varying light level is the most prominent change throughout the image sequence, although the environmental conditions also created varying object highlights, moving shadows, and many subtle contrast changes between the objects in the image. The car in the image is the object of interest. The auto-iris mechanism in the camera was functioning, which causes a similar appearance in the background foliage throughout the image sequence. Notice that even with the auto-iris capability built into the camera, there is still a wide variation in image characteristics across the image sequence. This variation requires the use of an adaptive segmentation approach to compensate for these changes.

To precisely evaluate the effectiveness of the adaptive image segmentation system, we exhaustively defined the segmentation quality surfaces for each frame. The segmentation quality surfaces were defined for preselected ranges of *maxmin* and *hsmooth* parameters. *Maxmin* values, which affect segmentation performance in a non-linear fashion, were sampled exponentially over a range of values from 100 to 471. Values near 100 were spaced closer together than values at the upper end of the range. *Hsmooth* values were sampled linearly using numbers between 1 and 63. By selecting 32 discrete values (5 bits of resolution) for each of these parameter ranges, the search space contained 1024 different parameter combinations.

4.1 Basic Experiments

The first set of experiments with the adaptive segmentation system was divided into two separate phases: 1) a training phase where the optimization capabilities of the genetic algorithm were measured, and 2) a testing phase where we evaluated the reduction in effort achieved by utilizing previous segmentation experience. The image data was separated into two halves, 10 images (1,3,...,19) for training and 10 images (2,4,...,20) for testing. During the training phase, seed populations were selected using random locations on the combined segmentation quality surface for each image. The genetic system was then invoked using the seed population for each image and the convergence rate of the process was measured. Each training image was processed 100 times, each with a different collection of random starting points. These results were combined to compute the average number of generations needed to optimize each surface. The genetic component used a local population size of 10, a crossover rate of 0.8, and mutation rate of 0.01. A crossover rate of

0.8 indicates that, on average, 8 out of 10 members of the population will be selected for recombination during each generation. The mutation rate of 0.01 implies that on average, 1 out of 100 bits is mutated during the crossover operation to insure diversity in the local population.

The stopping criteria for the genetic process contains three tests. First, since the global maximum for each segmentation quality surface was known a priori (recall that the entire surface was precomputed), the first stopping criteria was the location of a parameter combination with 95% segmentation quality or higher. In experiments where the entire surface is not precomputed, this stopping criteria would be discarded. Second, the process terminates if 3 consecutive generations produce a decrease in the average population fitness for the local population. Third, if 5 consecutive generations fail to produce a new maximum value for the average population fitness, the genetic process terminates. If any one of these three conditions is met, the processing of the current image is stopped and the maximum segmentation quality currently in the local population is reported.

Figure 6 shows the combined segmentation quality surfaces for the three training images (Frames 1, 13 and 19) shown in Figure 5. Note that due to the complexity of these surfaces, most commonly-used search techniques (as discussed in Section 2) would not be effective at optimizing the segmentation quality. The surface pairs in Figure 6 also summarize the search point movement for the training images. The movement of the points to highly fit areas of the segmentation quality surfaces is very evident. Figure 7 shows the initial and final segmentation results corresponding to surfaces in Figure 6. The results are obtained using the individual in the genetic population with maximum fitness. Note that the portion of the car that is extracted from the image is larger in the final results for each training image.

At the end of training phase, the final local population from each of the training images (1,3,...,19) was combined to create a global population of 100 individuals. From this global population, the 10 initial seed members of each local population for the testing images (2,4,...,20) were selected. The testing was performed in a parallel fashion; the final local population for each of the testing images was not placed back into the global population for these tests. The alternative approach to testing, which processes each frame in the outdoor imagery database in a sequential manner and integrates the results into the global population, will be discussed at the end of this section.

Figure 8 illustrates the initial seed population and final local population for the selected testing images (Frames 4, 10, and 20). When compared to the populations in Figure 6, these figures clearly indicate the high quality of both the initial and final populations for each image. Since the fitness of each seed population is based on previous segmentation experience, the genetic process is able to converge to the global maximum much faster during the testing phase. The initial and final segmentation results for the testing images are shown in Figure 9. The improved quality of the initial segmentation results during testing can be visually compared with the initial results acquired during training (Figure 7). Note that the final segmentation quality is approximately the same during both training and testing.

During the training experiments, the maximum number of generations was 13, the minimum number was 5, and the average number of generations was 9. By combining the information accumulated during training in the global population, the average number of generations was reduced from 9 during training to 3 during testing. The results for all 20

frames of the outdoor imagery are summarized in Figure 10. Noting that the average number of generations was reduced from 9 during training to 3 during testing, equivalent segmentation performance during testing represents a considerable improvement in the adaptive system's efficiency. On average, the adaptive segmentation system visits approximately 2.5% of the search space (i.e., ~ 2.5 generations) for the experiments described here.

Since there are no other known adaptive segmentation techniques in the computer vision field to compare our system with, we measured the performance of the adaptive image segmentation system relative to the set of default Phoenix segmentation parameters^{8,11} and a traditional optimization approach. The *default parameters* have been suggested after extensive amounts of testing by various researchers who developed the Phoenix algorithm.⁸ The parameters for *traditional approach* are obtained by manually optimizing the segmentation algorithm on the first image in the database and then utilizing that parameter set for the remainder of the experiments. This approach to segmentation quality optimization is currently standard practice in state-of-the-art computer vision systems. Figure 11 presents the comparison of these three approaches. The average segmentation quality for the adaptive segmentation technique was 95.8% (average of 100 experiments). In contrast, the performance of the default parameters was only 55.6% while the traditional approach provided 63.2% accuracy. As the figure shows, the performance of both of these alternative approaches was highly erratic throughout the sequence. Figure 12 illustrates the quality of the segmentation results associated with the adaptive system, the default parameters, and the traditional approach. Each result corresponds to the average segmentation performance produced by each technique for the first frame in the database. By comparing the extracted car region in each of these images, as well as the overall segmentation of the entire image, it is clear that the adaptive segmentation results are superior to the other methods.

4.2 Sequential Testing Experiment

To measure the improvement in efficiency achieved by immediately reusing segmentation experience, we also conducted a set of experiments to investigate the reduction in computational effort obtained by processing the images in a sequential rather than parallel manner. Two separate *sequential tests* were performed. The first test processed the images in their original order (e.g., frames 1,2,3,...,20). The second test altered the sequence of images to simulate a multi-day scenario where the frequency of image collection decreases to approximately one hour. Each group of images in the sequence (e.g., frames {1,5,9,12,16,20}, {3,7,11,14,18}, {2,6,10,13,17}, or {4,8,15,19}) was designed to represent a collection of images acquired on a different day. Thus, using the sequence of images described above, we have simulated a four day long collection of images.

For each of these tests, the genetic population of the first frame in the image sequence was randomly selected. Once the segmentation performance for that frame was optimized by the genetic algorithm, the final population from that image was used to create the initial global population. This global population was then used to select the seed population for subsequent frames in the image sequence. While the size of the global population remained below 100, the final collection of individuals from each successive image was added to the global population. After the size of the global population reached 100 individuals, the final populations from each image had to compete (based on fitness) with the current members of the global population.

4.2.1 Single Day Sequential Test - Figure 13 illustrates the performance of the system for the single day sequence. Through the first four frames, the number of generations per image decreases as the system obtains segmentation experience. Although the number of generations does increase at several points beyond the fourth frame, the overall trend of this plot does indicate a reduction in computational effort. This claim is evident by noting that the system optimizes the segmentation quality of 50% (10 out of 20) of these images using the information present in the global population. No iterations of the genetic cycle are necessary in these cases.

4.2.2 Multiple Day Sequential Test - Figure 14 presents the results for the multi-day simulation. The images in the first "day" (frames {1,5,9,12,16,20}) show a continually decreasing level of computational effort. In this test, since there is a wider separation between the initial images in the sequence, the number of generations required for the first few frames is higher. When the second sequence (frames {3,7,11,14,18}) is encountered, the effort increases temporarily as the adaptive process fills in the knowledge gaps present as a result of the differences between the images in each sequence. The image sequence for the third "day" (frames {2,6,10,13,17}) was handled with almost no effort by the genetic cycle. Finally, the fourth image sequence (frames {4,8,15,19}) requires no effort by the genetic cycle at all; each image is optimized by the information stored in the global population. Twelve of the twenty frames in this test were immediately optimized using the global population.

4.3 Comparison of the Adaptive System with Random Search

Several tests were performed to compare the optimization capabilities of the adaptive segmentation system with a simple random walk through the search space. This experiment used only the training images (1,3,...,19) from the outdoor image database so that the adaptive system would not benefit from the reuse of segmentation experience from one image to the next. The intent of this restriction was to measure the efficiency of the genetic algorithm in optimizing a complex surface. In addition, the stopping criteria for the adaptive system was simplified so that when a surface point with 95% segmentation quality or better was located, the optimization process would terminate. The random walk algorithm searched the segmentation quality surface by visiting points randomly and used the same 95% stopping criteria. Finally, in order to insure correctness of the results, each segmentation quality surface was optimized by each technique 100 times and the results are averaged to create the performance figures.

Figure 15 presents a comparison of the efficiency for the two techniques described above. The bars represent the total number of points visited on the surface using each technique for each of the images and the average number of points visited for each approach. As the average values show, the adaptive technique is far superior to the random walk approach. In addition, the average number of points visited by the adaptive approach is ~ 6.9% of the total number of points on the surface, compared to the earlier experiments where we processed ~ 2.5% of the surface, since we have not reused any segmentation experience gained from processing earlier images.

Figure 16 contrasts the segmentation quality achieved by the two techniques. Since the adaptive segmentation technique insures the achievement of a near global maximum for each image, we modified the random walk approach so that it would terminate after the same number of visited locations required by the adaptive technique. The maximum segmentation quality achieved by the random approach was then com-

pared with the adaptive system. On the average, the adaptive system achieved 99.3% segmentation quality after the number of segmentations shown in Figure 16. In comparison, the random walk achieved only 81.4% of the maximum quality for the same number of segmentations for each image.

4.4 The Effectiveness of the Reproduction and Crossover Operators

A number of tests were performed to demonstrate the effectiveness of the reproduction and crossover operators in the adaptive image segmentation system. The optimization capability of the pure genetic algorithm was compared with two variations of the genetic algorithm. The first variation of the pure genetic algorithm was implemented without a reproduction operator. Instead of reproducing individuals according to their fitness values, the algorithm selected the individuals at random for further genetic operator action with the restriction that any individual be selected only once. The second variation of the genetic algorithm simply skipped a crossover operator. To ensure that this approach generates about the same number of offsprings as the pure genetic algorithm, the mutation rate of this approach was increased to the crossover rate (0.8) of the genetic process. The stopping criteria for each technique is to locate a surface point with 95% or higher segmentation quality. In order to ensure correctness of the results each image was tested by each technique 100 times and the results were averaged to create the performance figures. Figure 17 presents the comparison of the optimization capability for three techniques. As the histograms show, the pure genetic algorithm results are much better than the results of the other two approaches for both the training and testing experiments. This demonstrates that the reproduction and crossover operators are critical for the success of genetic algorithms.

4.5 Hybrid Search Combining Genetic Algorithm and Hill Climbing

We also explored a hybrid search scheme for adaptive image segmentation. This scheme combines the global search technique (genetic algorithm) with the specialized local search technique (hill climbing). In this approach the genetic algorithm first finds the hills and the hill climber climbs them. The control allows the switching between the genetic algorithm and the hill climbing according to simple transition rules. The switch of control from the genetic algorithm to the hill climbing takes place when the genetic algorithm finds a new maximum point. The maximum point is passed to the hill climber as the starting point. The hill climber passes the control over to the genetic algorithm when it reaches a local maximum, a point that is better than all of its adjacent points. The local maximum point replaces the maximum point in the current population, which was the starting point for the hill climbing, and the genetic algorithm proceeds with the population. To find adjacent points in hill climbing, we used Hamming distance so that the points differ in one bit value from the given point in binary representation of points. To reduce the cost of evaluating all the adjacent points before making each move, our approach was designed to try alternatives until an uphill move was found. When the hill climbing process examines all the adjacent points by flipping each bit in the current search point without finding an uphill move, the current point is taken as a local maximum and the process passes the control to the genetic algorithm.

The genetic algorithm used here is the same as in the basic adaptive segmentation technique. The basic adaptive technique is also referred to as the "baseline experiments." The number of segmentations required by the genetic and hill

climbing algorithms are shown in Figure 18. Figure 19 compares the performance of the hybrid scheme and the baseline experiments. The hybrid scheme results surpassed the baseline results by reducing the number of segmentations required to optimize the segmentation quality in 8 (out of 10) training frames. On the average, during training there is 15.3% improvement in performance. Figure 20 compares the testing performance of the hybrid scheme with the baseline experiments. The hybrid scheme results are no better than the baseline results, because the training results supplied the testing seed points that are located in highly fit regions of the search space and these points can hardly be optimized by the hill climbing process.

In summary, the hybrid search process performs well for the training experiments which proceed with random starting points. However, it can not improve performance over the genetic algorithm alone for the testing experiments because the search proceeds with highly fit starting points.

4.6 Simultaneous Optimization of Local and Global Measures

When the local and global measures have to be optimized simultaneously, the problem becomes a multiobjective optimization problem. In multiobjective (or vector valued) optimization, the notion of optimality can be best explained using the concept of Pareto optimality.¹⁰ The key concept is the partially greater than relation between two vectors of the same dimension. Given two vectors x and y , then x is partially greater than y if each element of x is greater than or equal to the corresponding element of y and at least one element of x is strictly greater than the corresponding element of y . Under these conditions, we say that x dominates y or y is inferior to x . If a vector is not dominated by any other vector it is said to be nondominated or non-inferior, and the set of all nondominated vectors is called Pareto-optimal set. The goal of a search for optima in a vector-valued space is, then, locating Pareto-optimal set. Since, the goal of the search is a set of solutions, a genetic algorithm has a built-in advantage over other optimization techniques by working with a population of candidate solutions.

The genetic algorithm of the adaptive image segmentation system for the single objective function has been applied with some modifications and extensions to include multiobjective functions. First, the data structure for each individual has been changed to hold a vector-valued fitness, i.e., both a local quality measure and a global quality measure. Second, the reproduction procedure is modified to select a subpopulation of individuals for each dimension of the quality measure. The selection process is repeated for the number of dimensions of a vector and the size of subpopulations selected in each iteration is the population size divided by the vector size. The generation of new population procedure in this system becomes as follows. (1) Select subgroups of individuals using each dimension of the quality measure in turn, (2) Shuffle all the selected individuals, (3) Combine the individuals using crossover and mutation operators. This simple procedure ensures that any segmentation parameter which performs above average on any quality measure of image segmentation will likely be survived while also giving the appropriate selection preference to parameters that are above average on more than one quality measure.

The other major difference added in this implementation is the "dominate" procedure. The procedure "dominate" examines non-dominancy of each segmentation parameter by comparing it with all other parameters in a population after the image segmentation evaluation procedure has been applied. It should be noted that this non-dominancy test is

strictly local. Pareto's concept of non-dominancy implies comparison of a point to all other points in the search space, but our non-dominancy test is limited to the current population. While a locally dominated point is also globally dominated, the converse is not necessarily true. A segmentation parameter which is nondominated in one generation may be dominated by a parameter which may emerge in a later generation. The "dominate" test is still useful because the set of nondominated parameters in each generation represents the current best guess of the Pareto-optimal set that will be improved in the future generations.

The stopping criteria for the multiobjective optimization system contains two conditions. First, the process terminates if an utopian parameter set, i.e., the one whose both the local quality and the global quality are above a predefined threshold of acceptance, is located. The thresholds for acceptable segmentation is 90% of the best segmentation. This criterion is useful only when the best for each segmentation quality surface is known a priori. Second, the process terminates if both the average local quality and the average global quality of the populations are decreased for three consecutive generations or not improved for five consecutive generations. If either of these conditions is met, the segmentation of the current image is stopped and the nondominated parameter sets are represented as the current best estimates of the Pareto-optimal set.

In our experiments, the global quality measure is the weighted sum of edge-border coincidence and boundary consistency with each measure having equal weight. Similarly, the local quality measure is the weighted sum of pixel classification, object overlap and object contrast with each measure having equal weight. Since, the individual quality surfaces were defined exhaustively, the global quality surfaces and the local quality surfaces for these images could be computed using a weighted sum. Figure 21 shows the global quality and local quality surfaces for frame 1.

The experiment of the multiobjective optimization was designed similar to the combined segmentation quality optimization experiments. The average performance for the training images is shown in Figure 22. The small number of generations in Frame1 and Frame2 are caused by utopian points which perform well in both the global measure and the local measure. Frame 3 was selected to describe the complete experimental results of multiobjective optimization. Figure 23 illustrates the genetic search process at each generation by plotting the individuals in the population on the quality plane. The upper right corner of the plane represent the utopian point which has the maximum fitness values in both dimensions of the segmentation quality measure. The dark squares in the plots represent the locally nondominated points at each generation. The plotted planes have less than 10 points (the current population size), because some individuals in the population have the same fitness values and are plotted at the same location. Figure 23(c) displays the utopian point at the upper right corner, which caused the termination of the genetic search process after second generation. As in the single objective experiments, using the seed populations obtained from the global population, the genetic algorithm was applied to each testing image. Figure 24 compares the performance of the adaptive image segmentation system for multiobjective functions during the training and testing phases. As the average number of genetic cycles shows, the training results reduce the search efforts during testing. Figure 25 illustrates the quality of the genetic population at each generation for Frame 4, which is selected to describe the testing experiments. Since the fitness of the seed population is usually higher, the genetic process converges to the Pareto-optimal set much faster during the testing phase.

The intent of this experiment was to explore the applicability of the adaptive image segmentation system to the multiobjective functions. As the results show, it provides very promising performance for the simultaneous optimization of the global and local segmentation qualities.

5. CONCLUSIONS

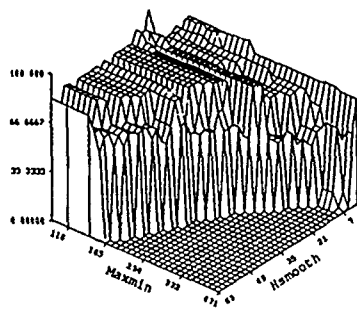
We have shown the ability of the adaptive image segmentation system to provide high quality (> 95%) segmentation results in a minimal number of segmentation cycles. The goal of this research was to perform adaptive image segmentation and evaluate the convergence properties of the closed-loop system using outdoor data. The performance improvement provided by the adaptive system was consistently greater than ~33% over the traditional approach or the default segmentation parameters.^{8,11} We also compared the performance of genetic algorithm with genetic and hill climbing combination and presented results on multiobjective optimization for local and global quality measures. In the future, we plan to use a data set with dramatic environmental variations and we will also utilize a larger number of segmentation parameters.

REFERENCES

1. B. Bhanu, S.K. Lee, and J.C. Ming, "Adaptive Image Segmentation Using A Genetic Algorithm," *Proceedings of DARPA Image Understanding Workshop*, pp. 1043-1055 (May, 1989).
2. K. DeJong, "Learning with Genetic Algorithms: An Overview," *Machine Learning* 3 pp. 121-138 (1988).
3. J.M. Fitzpatrick, J.J. Grefenstette, and D. Van Gucht, "Image Registration by Genetic Search," *Proceeding of IEEE Southeastern Conference*, pp. 460-464 (1984).
4. K.S. Fu and J.K. Mui, "A Survey on Image Segmentation," *Pattern Recognition* 13 pp. 3-16 (1981).
5. A.M. Gillies, "Machine Learning Procedures for Generating Image Domain Feature Detectors," Ph.D. Thesis, Dept. of Computer and Communication Sciences, University of Michigan (April, 1985).
6. R.M. Haralick and L.G. Shapiro, "Image Segmentation Techniques," *Computer Vision, Graphics, and Image Processing* 29 pp. 100-132 (1985).
7. J.H. Holland, "Escaping Brittleness: The Possibilities of General-Purpose Learning Algorithms Applied to Parallel Rule-Based Systems," pp. 593-623 in *Machine Learning: An Artificial Intelligence Approach, Vol. II*, ed. R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, Morgan Kaufmann Publishers, Inc. (1986).
8. K.I. Laws, "The Phoenix Image Segmentation System: Description and Evaluation," SRI International Technical Note No. 289 (December, 1982).
9. R. Ohlander, K. Price, and D.R. Reddy, "Picture Segmentation Using a Recursive Region Splitting Method," *Computer Graphics and Image Processing* 8 pp. 313-333 (1978).
10. J.D. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," *Proceedings of International Conference on Genetic Algorithms and Their Applications*, pp. 93-100 (1985).
11. S. Shafer and T. Kanade, "Recursive Region Segmentation by Analysis of Histograms," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1166-1171 (1982).

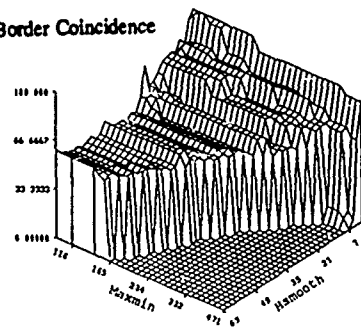


(a)

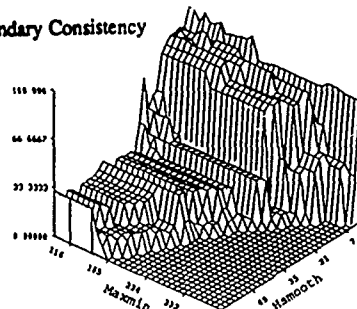


(b)

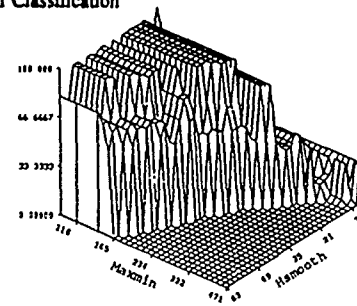
Edge-Border Coincidence



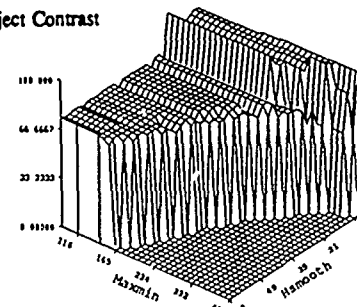
Boundary Consistency



Pixel Classification



Object Contrast



Object Overlap

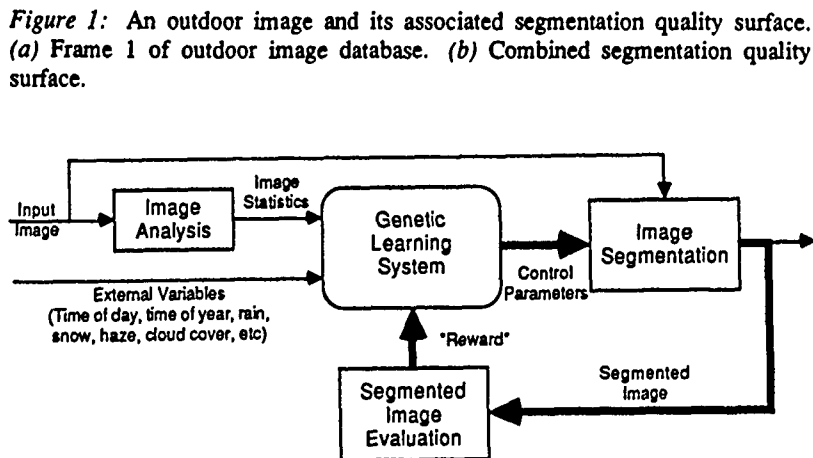
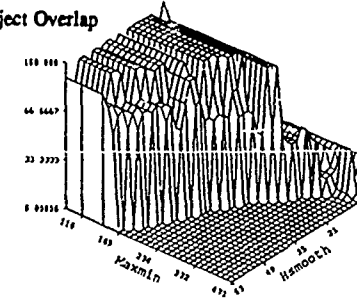


Figure 2: Block diagram of the adaptive image segmentation system.

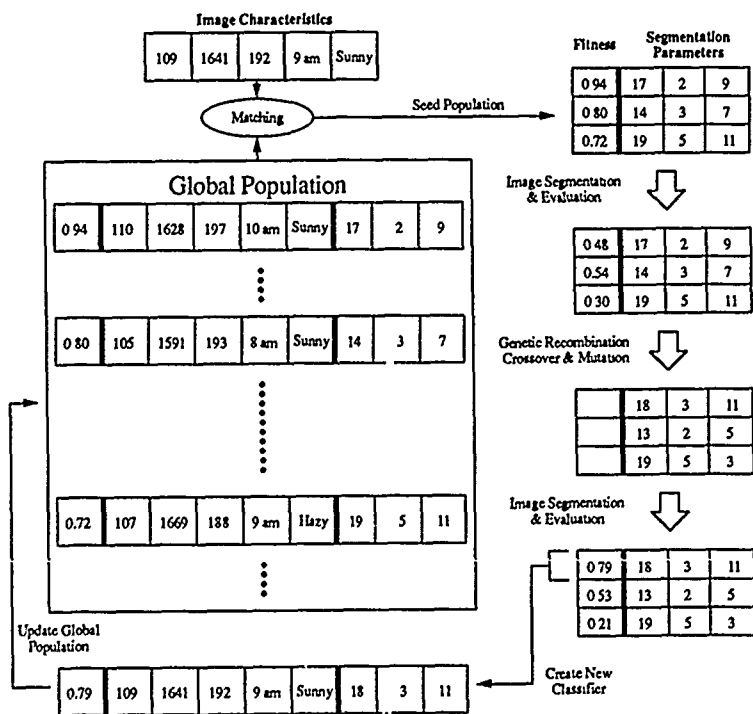


Figure 3: One complete cycle through the adaptive image segmentation

Figure 4: Individual quality surfaces



(a) Frame 1



(b) Frame 4



(c) Frame 10



(d) Frame 13

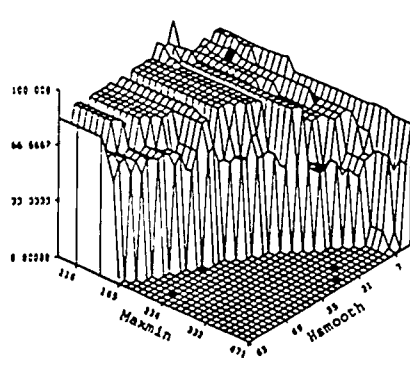


(e) Frame 19

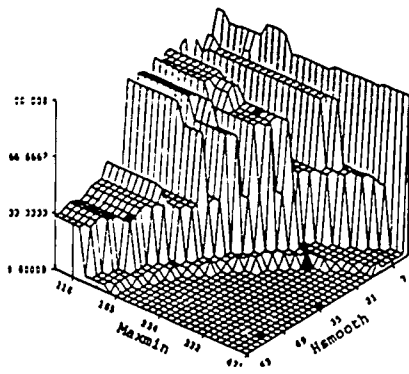


(f) Frame 20

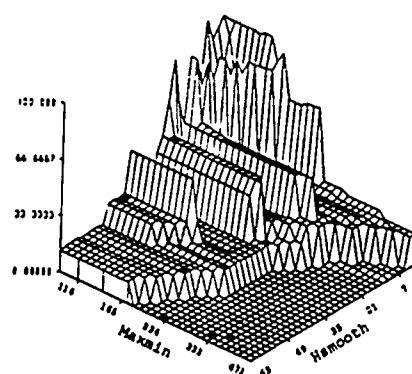
Figure 5: Selected color images from the outdoor experiments.



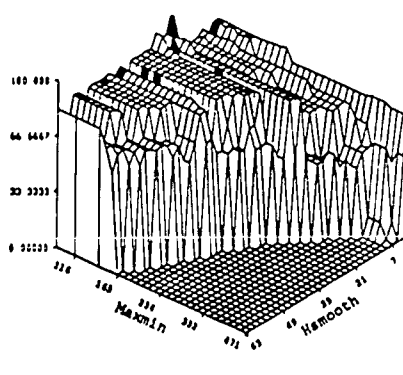
(a) Frame 1 Starting Locations



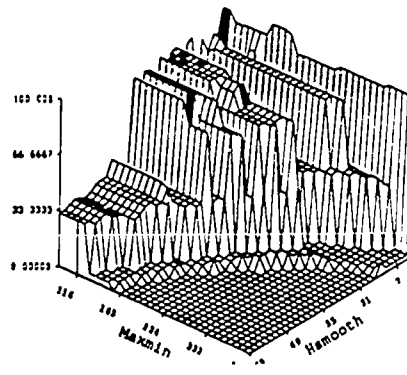
(c) Frame 13 Starting Locations



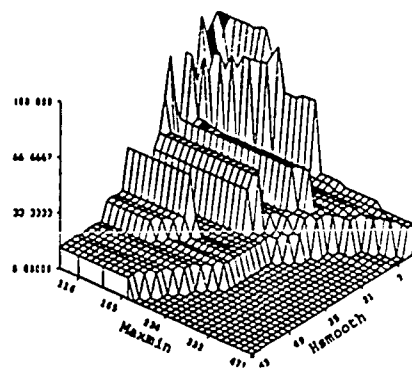
(e) Frame 19 Starting Locations



(b) Frame 1 Final Locations

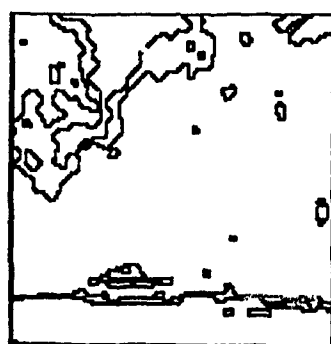


(d) Frame 13 Final Locations

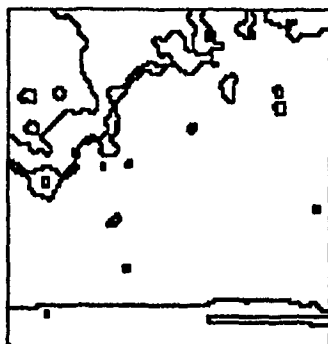


(f) Frame 19 Final Locations

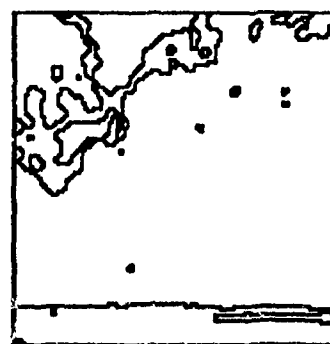
Figure 6: Starting and final search point locations for the training images.



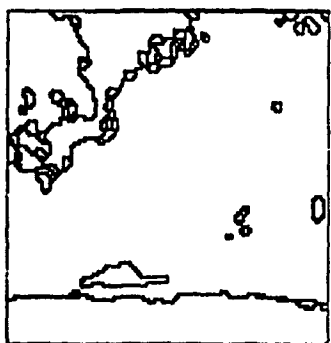
(a) Frame 1 Initial Segmentation



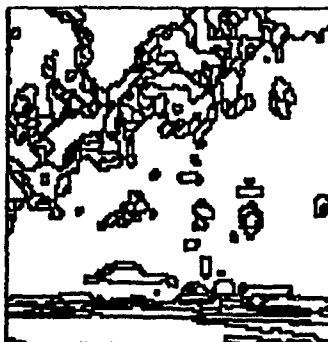
(c) Frame 13 Initial Segmentation



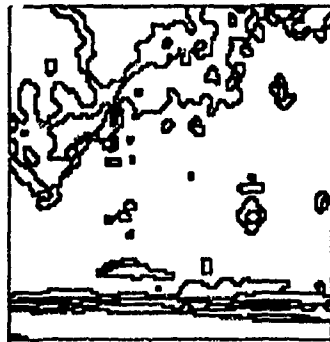
(e) Frame 19 Initial Segmentation



(b) Frame 1 Final Segmentation

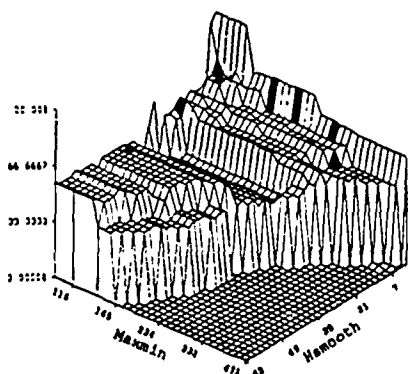


(d) Frame 13 Final Segmentation

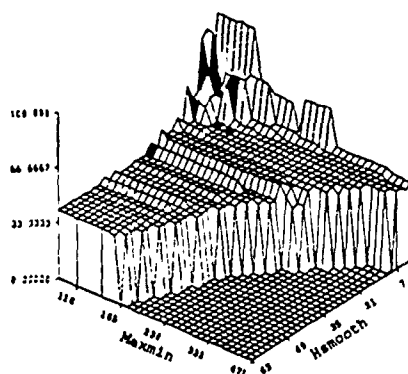


(f) Frame 19 Final Segmentation

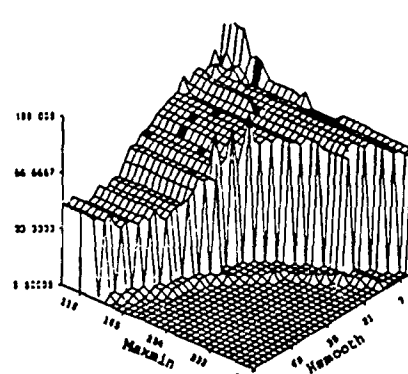
Figure 7: Initial and final segmentation results for the training images.



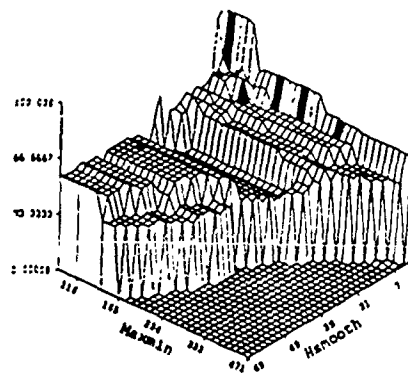
(a) Frame 4 Starting Locations



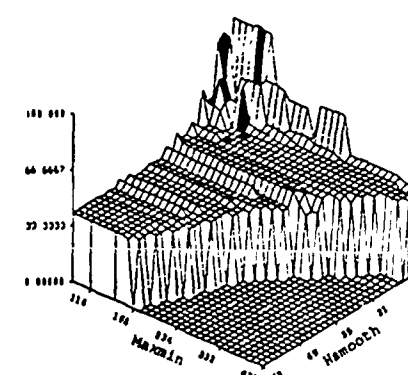
(c) Frame 10 Starting Locations



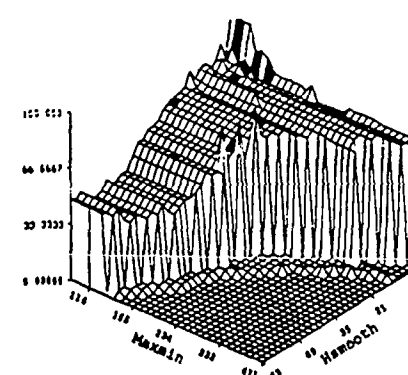
(e) Frame 20 Starting Locations



(b) Frame 4 Final Locations



(d) Frame 10 Final Locations



(f) Frame 20 Final Locations

Figure 8: Starting and final search point locations for the testing images.

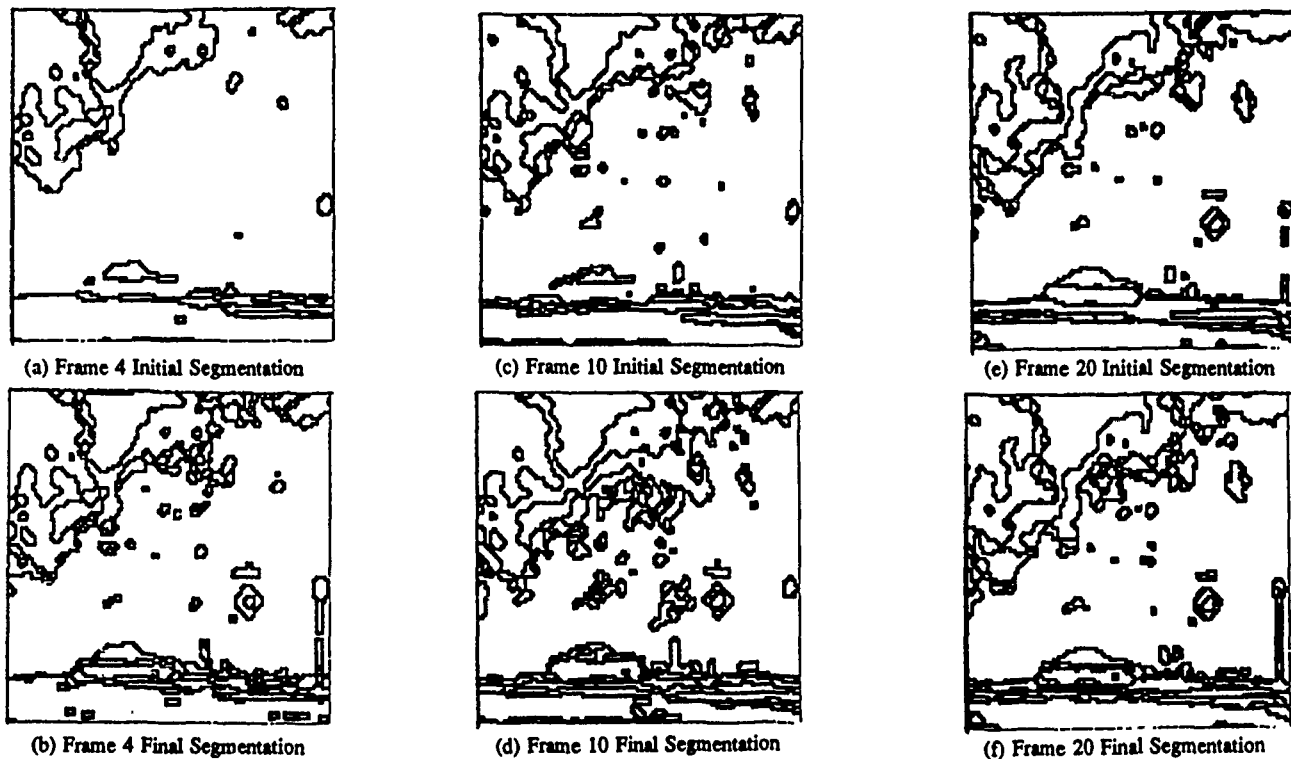


Figure 9: Initial and final segmentation results for the testing images.

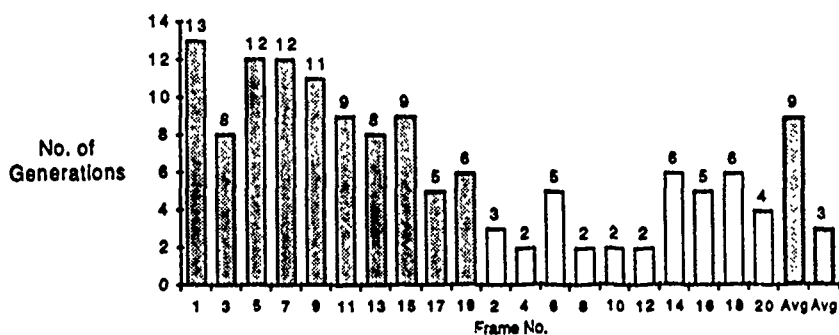


Figure 10: Performance comparison of the training and testing experiments on the outdoor imagery.

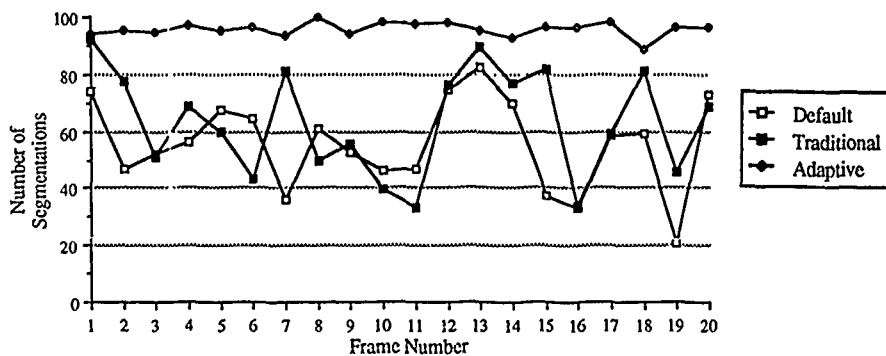
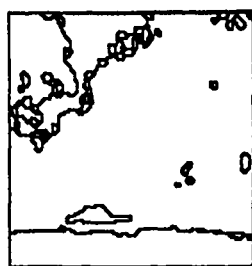
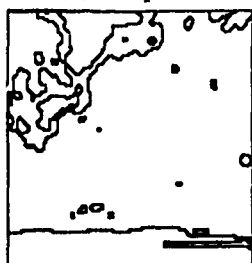


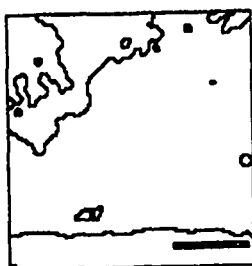
Figure 11: Comparison of the adaptive image segmentation system with default Phoenix parameters and the traditional segmentation approach commonly used in the computer vision field.



(a) Adaptive



(b) Default parameters



(c) Traditional Approach

Figure 12:
Segmentation performance comparison

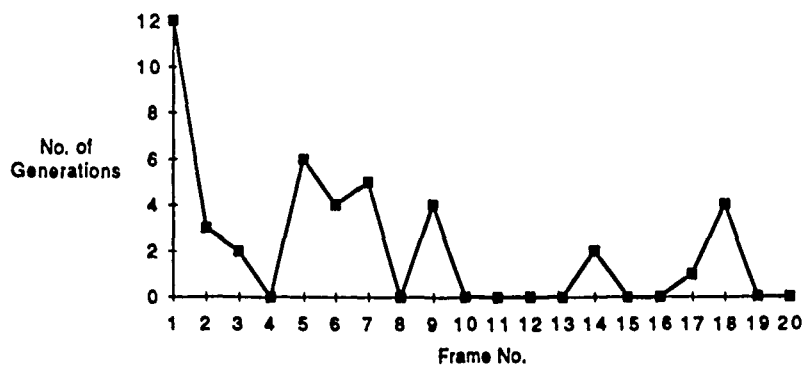


Figure 13: Performance results for the single day sequential test.

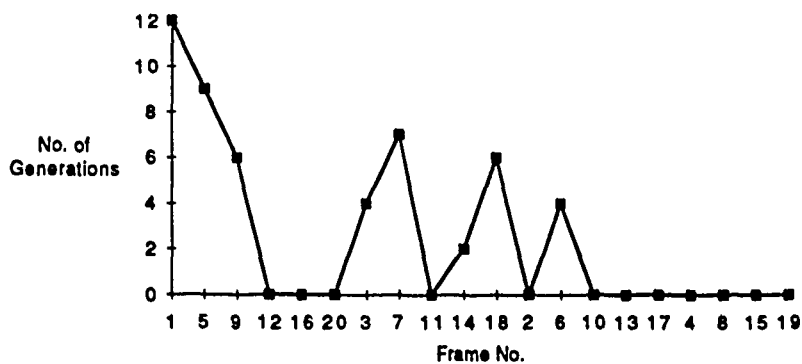


Figure 14: Performance results for the multiple day sequential test.

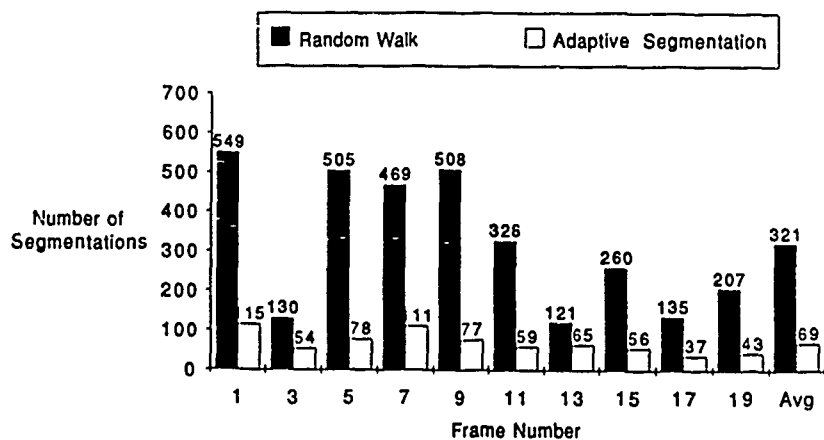


Figure 15: Performance comparison of the adaptive segmentation technique and a random walk approach.

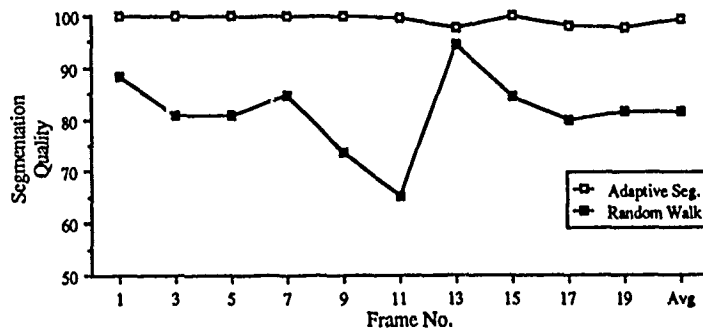
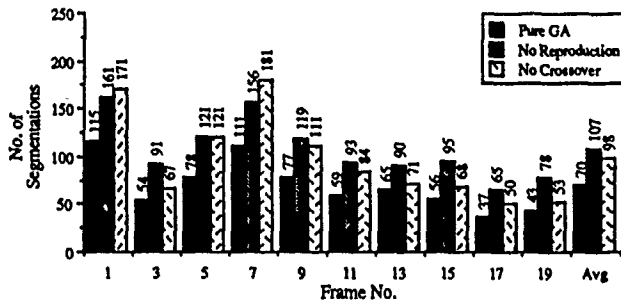
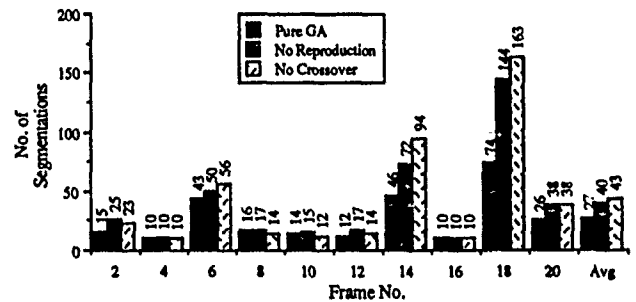


Figure 16: Segmentation quality performance for the adaptive segmentation technique and the random walk approach.



(a) Training Experiments



(b) Testing Experiments

Figure 17: Performance comparison of the pure genetic algorithm and its variations. The superior performance of the pure genetic algorithm demonstrates the effectiveness of the reproduction and crossover operators.

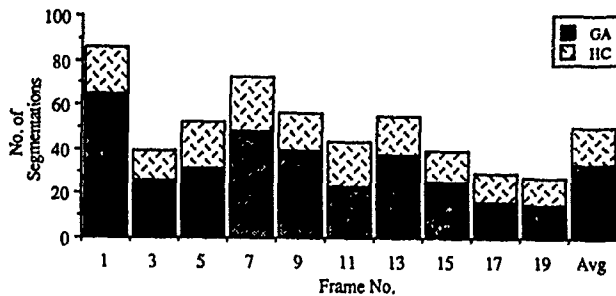


Figure 18: Performance summary for the hybrid scheme training experiments. The total number of segmentations required by the genetic algorithm and the hill climbing to optimize the segmentation quality of each image is indicated in the graph.

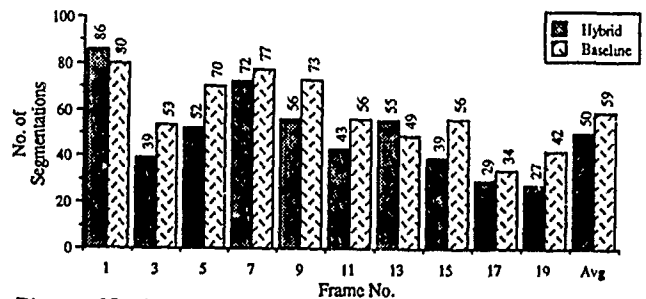


Figure 19: Performance comparison of the hybrid scheme and the baseline training experiments. By combining the genetic algorithm with the hill climbing, the hybrid scheme reduces the average computational effort needed to optimize the segmentation quality.

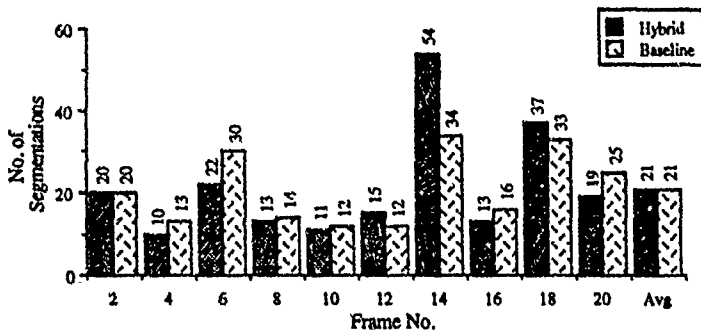


Figure 20: Performance comparison of the hybrid scheme and the baseline testing experiments. The hybrid scheme shows no performance improvement over the genetic algorithm alone because the training results provided highly fit seed points for the testing experiments.

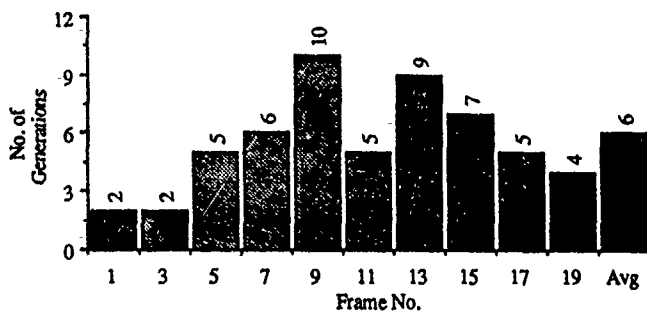
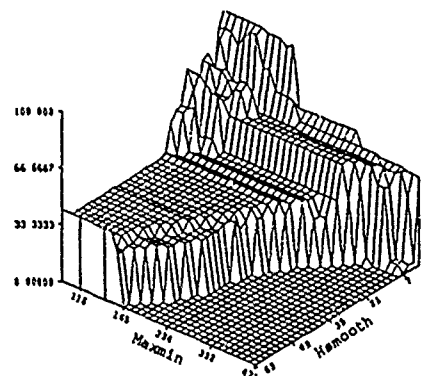
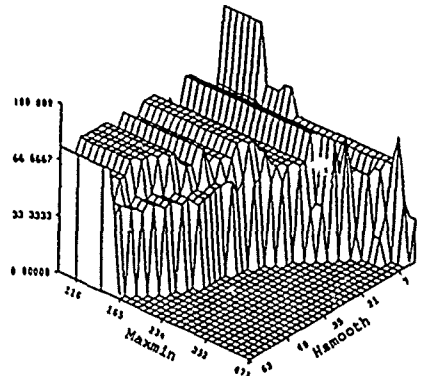


Figure 22: Performance summary of the training experiment for multiobjective optimization. Starting with random seed points on the quality surfaces, the adaptive image segmentation process optimized the global quality and the local quality of each image in the number of generations (average of 100 experiments) indicated in the graph.



(a) Global Quality



(b) Local Quality

Figure 21: Global and local quality surfaces for Frame 3

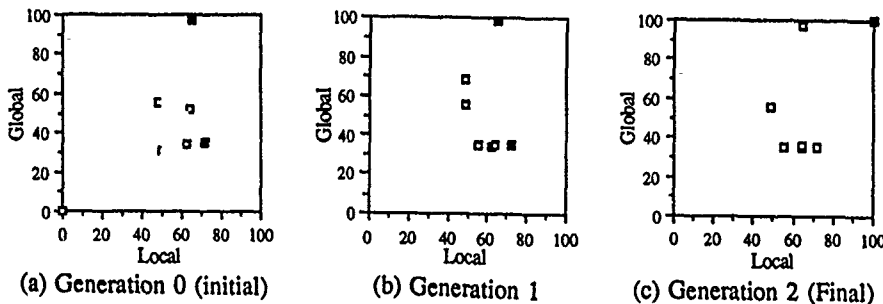


Figure 23: Search points plotted on the quality plane for Frame 3

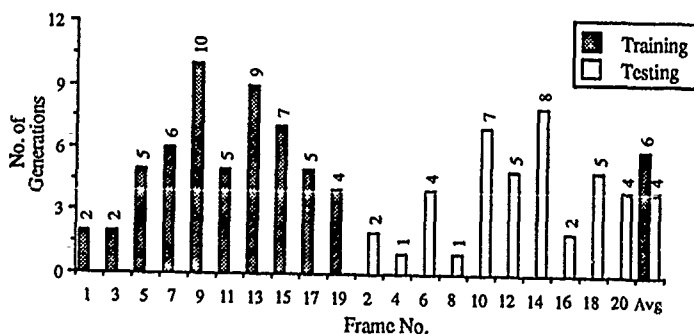


Figure 24: Performance comparison of the training and testing experiments for the multiobjective optimization.

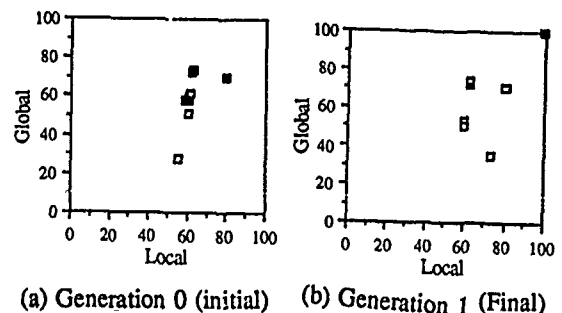


Figure 25: Search points plotted on the quality plane for Frame 4 during the testing experiments.

Extensions of a Theory of Networks for Approximation and Learning: dimensionality reduction and clustering

Tomaso Poggio and Federico Girosi

ABSTRACT

Learning an input-output mapping from a set of examples, of the type that many neural networks have been constructed to perform, can be regarded as synthesizing an approximation of a multi-dimensional function. From this point of view, this form of learning is closely related to regularization theory. The theory developed in Poggio and Girosi (1989) shows the equivalence between regularization and a class of three-layer networks that we call regularization networks. These networks are not only equivalent to generalized splines, but are also closely related to the classical Radial Basis Functions used for interpolation tasks and to several pattern recognition and neural network algorithms. In this note, we extend the theory by defining a general form of these networks which we call Hyper Basis Functions. They have two sets of modifiable parameters in addition to the coefficients c_i : moving centers and adjustable norm-weights. Moving the centers is equivalent to task-dependent clustering and changing the norm weights is equivalent to task-dependent dimensionality reduction.

1 Introduction

In previous papers (Poggio and Girosi, 1989, 1990) we have shown the equivalence between regularization and a class of three-layer networks that we called regularization networks and that are related to the classical interpolation technique of Radial Basis Functions.

Let $S = \{(\mathbf{x}_i, y_i) \in \mathbb{R}^n \times \mathbb{R} | i = 1, \dots, N\}$ be a set of data that we want to approximate by means of a function f . The regularization approach (Tikhonov, 1963; Tikhonov and Arsenin, 1977; Morozov, 1984; Bertero, 1986) selects the function f that solves the variational problem of minimizing the functional

$$H[f] = \sum_{i=1}^N (y_i - f(\mathbf{x}_i))^2 + \lambda \|Pf\|^2 \quad (1)$$

where P is a constraint operator (usually a differential operator), $\|\cdot\|$ is a norm on the function space to whom Pf belongs (usually the L^2 norm) and λ is a positive real number, the so called regularization parameter. The structure of the operator P , that is called "stabilizer", embodies the a priori knowledge about the solution, and therefore depends on the nature of the particular prob-

lem that has to be solved (for instance, it is not needed in the case of P corresponding to a Gaussian or bell-shaped Green's function). We have shown (Poggio and Girosi, 1989) that the solution of the variational problem (1) has the following simple form:

$$f(\mathbf{x}) = \sum_{i=1}^N c_i G(\mathbf{x}; \mathbf{x}_i) + p(\mathbf{x})$$

where $G(\mathbf{x})$ is the Green's function (Stakgold, 1979) of the self-adjoint differential operator $\hat{P}P$, \hat{P} being the adjoint operator of P , $p(\mathbf{x})$ is a linear combination of functions that span the null space of P , and the coefficients c_i satisfy a linear system of equations that depend on the N "examples", i.e. the data to be approximated. The form of the term $p(\mathbf{x})$ depends on the stabilizer that has been chosen and on the boundary conditions, and therefore on the particular problem that has to be solved. For this reason, and since its inclusion does not modify the main conclusions, we will disregard it in the following. If P is an operator with radial symmetry, the Green's function G is radial and therefore the approximating function becomes:

$$f(\mathbf{x}) = \sum_{i=1}^N c_i G(\|\mathbf{x} - \mathbf{x}_i\|^2), \quad (2)$$

which is a sum of radial functions, each with its center \mathbf{x}_i on a distinct data point. Thus the number of radial functions, and corresponding centers, is the same as the number of examples.

In this note we indicate how to extend the technique into three natural directions:

1. The computation of a solution of the form (2) has a complexity (number of radial functions) that is independent of the dimensionality of the input space but is on the order of the dimensionality of the training set (number of examples), which is usually high. We show how to justify in terms of the regularization framework an approximation of equation (2) in which the number of centers is much smaller than the number of examples and the positions of the centers are modified during learning (Poggio and Girosi, 1989). The key idea is to consider a specific form of an approximation to the solution of the standard regularization problem. Moving centers

are equivalent to the free knots of nonlinear splines. In the context of networks they were first suggested as a potentially useful heuristics by Broomhead and Lowe (1988) and used by Moody and Darken (1989).

2. It is natural to try to extend the form of the solution (2) by considering the superposition of different types of Green's functions (Poggio and Girosi, 1989, 1990a) (for example basis functions of different scales). This extension is natural within the framework of regularization (and has a direct Bayesian interpretation) by considering a more general functional than equation (1) containing several stabilizers. We will show how the well-defined but underconstrained variational problem associated with the new functional can be transformed into an overconstrained problem.
3. In equation (2) the norm $\|x - x_i\|$ may be considered as a *weighted norm*

$$\|x - x_i\|_W^2 = (x - x_i)^T W^T W (x - x_i)$$

where W is a square matrix and the superscript T indicates the transpose. In the simple case of diagonal W the diagonal elements w_{ii} assign a specific weight to each input coordinate, and the standard Euclidean norm is obtained when W is set to the identity matrix. They play a critical role whenever different types of inputs are present. We will show how the weighted norm idea can be derived from a slightly more general functional than equation (1). The associated variational problem is well-defined but underconstrained; it can be transformed into an overconstrained problem by using a certain approximation technique.

We call *Hyper Basis Functions*, in short *HyperBFs*, the most general form of regularization networks based on these three extensions.

2 Moving Centers

The solution given by standard regularization theory to the approximation problem can be very expensive in computational terms when the number of examples is very high. The computation of the coefficients of the expansion can become then a very time consuming operation: its complexity grows polynomially with N , (roughly as N^3) since an $N \times N$ matrix has to be inverted. In addition, the probability of ill-conditioning is higher for larger and larger matrices (it grows like N^3 for a $N \times N$ uniformly distributed random matrix) (Demmel, 1987). We now show a way to reduce the complexity of the problem, introducing an approximation to the regularized solution. While the exact regularization solution is equivalent to generalized splines with *fixed* knots, the approximated solution is equivalent to generalized splines with *free* knots.

2.1 An approximation to the regularization solution

A standard technique, sometimes known as Galerkin's method, that has been used to find approximate solutions of variational problems, is to expand the solution

on a finite basis. The approximated solution $f^*(x)$ has then the following form:

$$f^*(x) = \sum_{i=1}^n c_i \phi_i(x) \quad (3)$$

where $\{\phi_i\}_{i=1}^n$ is a set of linearly independent functions (Mikhlin, 1965). The coefficients c_i are usually found according to some rule that guarantees a minimum deviation from the true solution. In the case of standard regularization, when the functional to minimize is given by equation (1), this method gives the *exact* solution if n is equal to the number of data points N , and $\{\phi_i\}_{i=1}^n = \{G(x; x_i)\}_{i=1}^N$, where G is the Green's function of the operator \hat{P} . In this case the unknown coefficients of the expansion (3) can be obtained in a simple way by substituting expansion (3) in the regularization functional (1), that becomes a function $H[f^*] = H^*(c_1, \dots, c_N)$, and then by minimizing $H[f^*]$ with respect to the coefficients, that is by setting:

$$\frac{\partial H[f^*]}{\partial c_i} = 0 \quad i = 1, \dots, N. \quad (4)$$

It can be easily shown (Poggio and Girosi, 1989) that, if the Green's function vanishes on the boundary of the region that is considered, the set of equations (4) is a linear system whose solution gives the standard regularization coefficients. In more general cases the basis $\{\phi_i\}_{i=1}^n$ should be enlarged, to include terms that generate the null space of P , in order to obtain the correct solution. For simplicity, we disregard these terms in the following, since they do not change the main conclusions. A natural approximation to the exact solution will be then of the form:

$$f^*(x) = \sum_{\alpha=1}^n c_{\alpha} G(x; t_{\alpha}) \quad (5)$$

where the parameters t_{α} , that we call "centers", and the coefficients c_{α} are unknown, and are in general fewer than the data points ($n \leq N$). This form of solution has the desirable property of being an universal approximator for continuous functions (Girosi and Poggio, 1989) and to be the only choice that guarantees that in the case of $n = N$ and $\{t_{\alpha}\}_{\alpha=1}^n = \{x_i\}_{i=1}^N$ the correct solution (of equation 1) is consistently recovered. We will see later in section (5) how to find the unknown parameters of this expansion.

3 Different types of Basis Functions.

This scheme can be further extended by considering in equation (5) the superposition of different types of functions G , such as Gaussians at different scales.

The function f to be approximated is regarded as the sum of p components f^m , $m = 1, \dots, p$, each component having a different prior probability. This assumption is clearly meaningful only if $p \ll N$. Therefore the functional $H[f]$ to minimize will contain p stabilizers P^m , p regularization parameters λ_m , and will be written as

$$H[f] = \sum_{i=1}^N (y_i - \sum_{m=1}^p f^m(\mathbf{x}_i))^2 + \sum_{m=1}^p \lambda_m \|P^m f^m\|^2. \quad (6)$$

The Euler-Lagrange equations associated with equation (6) have the form:

$$\hat{P}^m P^m f^m(\mathbf{x}) = \frac{1}{\lambda_m} \sum_{i=1}^N (y_i - \sum_{k=1}^p f^k(\mathbf{x}_i)) \delta(\mathbf{x} - \mathbf{x}_i) \quad (7)$$

for $m = 1, \dots, p$. As in the case of standard regularization, the solution of equation (7) is a linear superposition of Green's functions:

$$f^m(\mathbf{x}) = \sum_{i=1}^N c_i^m G^m(\mathbf{x}; \mathbf{x}_i). \quad (8)$$

The function $F(\mathbf{x})$ that minimizes the functional $H[f]$ is then a linear superposition of linear superpositions of the Green's functions G^m corresponding to the stabilizers P^m , that is

$$F(\mathbf{x}) = \sum_{m=1}^p \sum_{i=1}^N c_i^m G^m(\mathbf{x}; \mathbf{x}_i) + p(\mathbf{x}), \quad (9)$$

where $p(\mathbf{x})$ is a linear combination of functions that span the null spaces of the stabilizers. For instance, when $G^m(\mathbf{x})$ are Gaussian a polynomial is not needed, though it can always be added. For other Green's functions the theory requires an appropriate $p(\mathbf{x})$.

Substitution of equation (8) in equation (7) yields a linear system for the coefficients c_i^m . There is a simple relation between the coefficients associated to two different stabilizers, that is

$$c_i^m \lambda_m = c_i^n \lambda_n, \quad i = 1, \dots, N; \quad n, m = 1, \dots, p.$$

This means that if a component $f^m(\mathbf{x})$ of the solution is given, the other $p-1$ ones can be recovered by a simple scaling operation. This is expected, since the underlying variational problem is *underconstrained*: we are trying to obtain Np coefficients from a set of N data points. The form of the solution (9) is appealing: if all the coefficients c_i^m were independent and free to vary, the system could "choose" among different stabilizers, depending on the site. In order to retain the form (9) of the solution, while making the problem *overconstrained* instead of *underconstrained*, we choose a solution of the approximation problem of the following form (instead of equation 9):

$$\tilde{F}(\mathbf{x}) = \sum_{m=1}^p \tilde{f}^m(\mathbf{x}) + p(\mathbf{x}), \quad (10)$$

$$\tilde{f}^n(\mathbf{x}) = \sum_{\alpha=1}^{K_n} c_{\alpha}^n G^m(\mathbf{x}; \mathbf{t}_{\alpha}^n) \quad (11)$$

where $(1+d) \sum_{m=1}^p K_m < N$ and the coefficients c_{α}^m and the centers \mathbf{t}_{α}^m are unknowns. They can be found with a technique similar to the one described in section (5). Notice that equations (10) and (11) are of the same form as equation (5) and share its approximation properties.

3.1 Multiple Scales.

This method leads in particular to radial basis functions of multiple scales for the reconstruction of the function f . Suppose we know *a priori* that the function to be approximated has components on a number p of scales $\sigma_1, \dots, \sigma_p$: we can use this information to choose a set of p stabilizers whose Green's functions are, for example, Gaussians of variance $\sigma_1, \dots, \sigma_p$. We have (Poggio and Girosi, 1989, 1990a):

$$\|P^m f^m\|^2 = \sum_{k=0}^{\infty} a_k^m \int_{R^n} d\mathbf{x} (D^k f^k(\mathbf{x}))^2$$

where $D^{2k} = \vec{\nabla}^{2k}$, $D^{2k+1} = \vec{\nabla} \vec{\nabla}^{2k}$ and $a_k^m = \frac{\sigma_m^{2k}}{k! 2^k}$, $\vec{\nabla}$ being the gradient operator. As a result, the solution will be a *superposition of superpositions* of Gaussians of different variances. Of course, the Gaussians with large σ should be preset, depending on the nature of the problem, to be fewer and therefore on a sparser grid, than the Gaussians with a small σ .

The HyperBF method also yields non-radial Green's functions - by using appropriate stabilizers - and also Green's functions with a lower dimensionality - by using the associated f^m and P^m in a suitable lower-dimensional subspace. Again this reflects *a priori* information that may be available about the nature of the mapping to be learned. In the latter case the information is that the mapping is of lower dimensionality or has lower dimensional components.

4 Weighted norm

The norm in equation (5) is usually intended as an Euclidean norm. If the components of \mathbf{x} are of different types, it is natural to consider a *weighted norm* defined as

$$\|\mathbf{x}\|_{\mathbf{W}}^2 = \mathbf{x}^T \mathbf{W}^T \mathbf{W} \mathbf{x},$$

since the relative scale of the components is otherwise arbitrary. The case in which the matrix \mathbf{W} is known (from prior information) does not present any difficulty. It is interesting, however, to see what it means in terms of the underlying regularization principle.

4.1 Weighted norm and regularization

The regularization principle consists in finding the f that minimizes the functional:

$$H_{\mathbf{W}}[f] = \sum_{i=1}^N (y_i - f(\mathbf{x}_i))^2 + \lambda \|Pf\|_{\mathbf{W}}^2 \quad (12)$$

where we assume that P is radially symmetric in the variable \mathbf{y} and that $\mathbf{y} = \mathbf{W}\mathbf{x}$ (i.e. \mathbf{y} is a known linear transformation of \mathbf{x} that depends on the parameters \mathbf{W}). This means that the smoothness constraint is given in a space that is an affine transformation of the original \mathbf{x} space. The Green's function associated with equation (12) is

$$G(\|\mathbf{y}\|^2) = G(\|\mathbf{x}\|_{\mathbf{W}}^2) \quad (13)$$

with $\|x\|_W^2 = x^T W^T W x$.

Suppose now that the parameters W are unknown. We can formulate the problem of finding f and W that minimize the functional $H_W(f)$. Notice that the relevant quantity is $M = W^T W$, since W only appears in this form. The matrix M is symmetric and positive definite; it has therefore a unique, symmetric "square root" R , such that $M = R^T R = R^2$. One could choose to identify W with R . W would be therefore symmetric, with $\frac{(d^2+d)}{2}$ independent parameters.

Thus finding the optimal W corresponds to finding the best stabilizer among those that are expressed in a coordinate system which is a linear transformation of the original one. The parameters W of the linear transformation become parameters of H with respect to which the functional is minimized.

The simplest case is the case of W diagonal and $G(x) = e^{-x^2}$. In this case

$$G(\|x\|_W^2) = e^{-x_1^2 w_1^2} e^{-x_2^2 w_2^2} \dots e^{-x_n^2 w_n^2}$$

and thus the components w_i of W are equivalent to the inverse of the variance σ of each component of the multidimensional Gaussian.

In the probabilistic interpretation of standard regularization (see Poggio and Girosi, 1989) the term $\lambda \|Pf\|^2$ in the regularization functional corresponds to the following prior probability in a Bayesian formulation in which the MAP (Maximum A Posteriori) estimate is sought:

$$Prob(f) = e^{-\lambda \|Pf\|^2}.$$

Our extension corresponds to choosing the stabilizer $P_W = \|Pf(y)\|^2$, with $y = Wx$. The stabilizer P_W is parametrized by the matrix W and defines a prior $Prob_W(f)$ which is also parametrized by W .

The solution of the variational problem (12) has the form

$$f(x) = \sum_{i=1}^N c_i G(\|x - x_i\|_W^2), \quad (14)$$

where the coefficients c_i and the elements of the matrix W must be estimated. Here again we are facing an underconstrained variational problem, since we are trying to determine $N + \frac{(d^2+d)}{2}$ parameters from N data points. The same considerations of section (3) apply: in order to transform the problem into an overconstrained problem, we look for a solution of the form

$$f^*(x) = \sum_{\alpha=1}^n c_\alpha G(\|x - t_\alpha\|_W^2) \quad (15)$$

5 How to learn centers' positions and norm weights

Suppose that we look for an approximated solution of the regularization problem of the form (15). We now have the problem of finding the n coefficients c_α , the $d \times n$ coordinates of the centers t_α and the $\frac{(d^2+d)}{2}$ elements of the matrix M so that the expansion (12) is optimal.

To avoid too many indices, we will only consider here the case $p = 1$ in eq. 10. The extension is obvious. In this case we can use the natural definition of optimality given by the functional H . We then impose the condition that the set $\{c_\alpha, t_\alpha | \alpha = 1, \dots, n\}$ and the matrix M must be such that they minimize $H[f^*]$, and the following equations must be satisfied:

$$\frac{\partial H[f^*]}{\partial c_\alpha} = 0, \quad \frac{\partial H[f^*]}{\partial t_\alpha} = 0, \quad \frac{\partial H[f^*]}{\partial M} = 0, \quad \alpha = 1, \dots, n.$$

Gradient-descent is probably the simplest approach for attempting to find the solution to this problem, though, of course, it is not guaranteed to converge. Several other iterative methods, such as versions of conjugate gradient and simulated annealing (Kirkpatrick et al., 1983) may be more efficient than gradient descent and should be used in practice. Since the function $H[f^*]$ to minimize is in general non-convex, a stochastic term in the gradient descent equations may be advisable to avoid local minima. In the stochastic gradient descent method the values of c_α , t_α and M that minimize $H[f^*]$ are regarded as the coordinates of the stable fixed point of the following stochastic dynamical system:

$$\dot{c}_\alpha = -\omega \frac{\partial H[f^*]}{\partial c_\alpha} + \eta_\alpha(t), \quad \alpha = 1, \dots, n$$

$$\dot{t}_\alpha = -\omega \frac{\partial H[f^*]}{\partial t_\alpha} + \mu_\alpha(t), \quad \alpha = 1, \dots, n$$

$$\dot{M} = -\omega \frac{\partial H[f^*]}{\partial M} + \Omega(t)$$

where $\eta_\alpha(t)$, $\mu_\alpha(t)$ and $\Omega(t)$ are white noise of zero mean and ω is a parameter determining the microscopic timescale of the problem and is related to the rate of convergence to the fixed point. Defining

$$\Delta_i \equiv y_i - f^*(x) = y_i - \sum_{\alpha=1}^n c_\alpha G(\|x_i - t_\alpha\|_W^2)$$

and setting $\lambda = 0$ for simplicity (the more general case can be approached in a similar way) in equation (1) we obtain

$$H[f^*] = H_{c,t,M} = \sum_{i=1}^N (\Delta_i)^2.$$

The important quantities - that can be used in more efficient schemes than gradient descent - are, with

$$\|x_i - t_\alpha\|_W^2 = (x_i - t_\alpha)^T M (x_i - t_\alpha)$$

and $M = W^T W$:

- for the c_α

$$\frac{\partial H[f^*]}{\partial c_\alpha} = -2 \sum_{i=1}^N \Delta_i G(\|x_i - t_\alpha\|_W^2); \quad (16)$$

- for the centers t_α

$$\frac{\partial H[f^*]}{\partial t_\alpha} = 4c_\alpha \sum_{i=1}^N \Delta_i G'(\|x_i - t_\alpha\|_{\mathbf{W}}^2) \mathbf{M}(x_i - t_\alpha) \quad (17)$$

- and for \mathbf{M}

$$\frac{\partial H[f^*]}{\partial \mathbf{M}} = -2 \sum_{\alpha=1}^n c_\alpha \sum_{i=1}^N \Delta_i G'(\|x_i - t_\alpha\|_{\mathbf{W}}^2) Q_{i,\alpha} \quad (18)$$

where $Q_{i,\alpha} = (x_i - t_\alpha)(x_i - t_\alpha)^T$ is a dyadic product and G' is the first derivative of G .

Remarks

1. Instead of equation (18) for \mathbf{M} the following equation can be used for \mathbf{W} :

$$\frac{\partial H[f^*]}{\partial \mathbf{W}} = -4\mathbf{W} \sum_{\alpha=1}^n c_\alpha \sum_{i=1}^N \Delta_i G'(\|x_i - t_\alpha\|_{\mathbf{W}}^2) Q_{i,\alpha} \quad (19)$$

2. From equation (18) the matrix \mathbf{M} is guaranteed to remain symmetric in a deterministic gradient descent scheme, since the right hand-side of the equation is symmetric (because the $Q_{i,\alpha}$ are correlation matrices and a linear combination of symmetric matrices is symmetric). Of course, the initial value must be a symmetric matrix and in the stochastic update scheme, the noise term must not break the symmetry. The matrix \mathbf{M} must satisfy the additional constraint of remaining positive definite (since the scalar product $x^T \mathbf{M} x$ must be non-negative). We conjecture that equations (16), (17) and (18) conserve the positive definiteness of \mathbf{M} if G is positive definite.
3. Equation (16) has a simple interpretation: the correction is equal to the sum over the examples of the products between the error on that example and the "activity" of the "unit" that represents with its center that example. Notice that $H[f^*]$ is quadratic in the coefficients c_α , and if the centers and the matrix \mathbf{M} are kept fixed, it can be shown (Poggio and Girosi, 1989) that the optimal coefficients are given by

$$c = (G^T G + \lambda g)^{-1} G^T y \quad (20)$$

where we have defined $(y)_i = y_i$, $(c)_\alpha = c_\alpha$, $(G)_{i\alpha} = G(x_i; t_\alpha)$ and $(g)_{\alpha\beta} = G(t_\alpha; t_\beta)$. If λ is let go to zero, the matrix on the right side of equation (20) converges to the pseudoinverse of G (Albert, 1972), and if the Green's function is radial the approximation method of Broomhead and Lowe (1988) is recovered.

4. Equation (17) is similar to task-dependent clustering (Poggio and Girosi, 1989). This can be best seen

by assuming that Δ_i are constant: then the gradient descent updating rule makes the centers move as a function of the majority of the data, that is of the position of the clusters. In this case a technique similar to the k-means algorithm is recovered (MacQueen, 1967; Moody and Darken, 1989). Equating $\frac{\partial H[f^*]}{\partial t_\alpha}$ to zero we notice that, when the matrix \mathbf{M} is set to the identity matrix, the optimal centers t_α satisfy the following set of nonlinear equations:

$$t_\alpha = \frac{\sum_i P_i^\alpha x_i}{\sum_i P_i^\alpha} \quad \alpha = 1, \dots, n$$

where $P_i^\alpha = \Delta_i G'(\|x_i - t_\alpha\|^2)$. The optimal centers are then a weighted sum of the data points. The weight P_i^α of the data point i for a given center t_α is high if the interpolation error Δ_i is high there and the radial basis function centered on that knot changes quickly in a neighborhood of the data point. This observation suggests faster update schemes, in which a suboptimal position of the centers is first found and then the c_α are determined, similarly to the algorithm developed and tested successfully by Moody and Darken (1989).

5. Equation (19) (by assuming that

$$\sum_{\alpha=1}^n c_\alpha \Delta_i G'(\|x_i - t_\alpha\|_{\mathbf{W}}^2)$$

is asymptotically constant (!)) contains the quantity $\sum_{i=1}^N Q_{i,\alpha}$ which is an estimate of the correlation matrix of all the examples relative to t_α (modulus a normalization factor). Let us define $C_{m,\alpha}$ as the $d \times m$ matrix whose columns are the vectors of the examples $x_1 - t_\alpha, \dots, x_m - t_\alpha$. Then $\sum_{i=1}^N Q_{i,\alpha}$ can be written as $\sum_{i=1}^N Q_{i,\alpha} = C_{N,\alpha} C_{N,\alpha}^T$ and is the $d \times d$ correlation matrix (d being the number of components of x). Interestingly, in this case, equation (19), when inserted in the gradient descent equation, has the form:

$$\dot{\mathbf{W}} = -\mathbf{W} \mathbf{Q}$$

which has the solution

$$\mathbf{W}(t) = \mathbf{W}(0) e^{-\mathbf{Q}t} = \mathbf{W}(0) \sum_{j=1}^N e^{-\lambda_j t} \mathbf{e}_j \mathbf{e}_j^T$$

where \mathbf{e}_j are the eigenvectors of \mathbf{Q} and λ_j are the associated eigenvalues. All eigenvectors will decay to 0, the ones with the largest eigenvalues fastest. Since in the full equation the other terms such as Δ_i will keep \mathbf{W} from decaying to 0, we may expect that \mathbf{W} will converge to a matrix with rows that are similar to the eigenvectors of \mathbf{Q} with the smallest eigenvalues. In other words, the equation should converge to rows of \mathbf{W} that span the space orthogonal to the space spanned by the principal components of the input examples (i.e. the eigenvectors of \mathbf{Q} with the largest eigenvalues). In this case, the

matrix M is a projection operator that projects x into a space orthogonal to the space of the principal components. The principal components are the singular vectors of X , with the property that they span a nested set of optimal subspaces. This interpretation of the gradient descent equation is just a rough indication of what may happen, because of the very strong underlying assumptions. It turns out that in the object recognition case (Poggio and Edelman, 1990), the interpretation is perfectly consistent with what one expects, given the (linear) computational theory underlying the problem (Basri and Ullmann, 1990; see also the appendix in Edelman and Poggio, 1990). Under orthographic projection, the vectors representing views of the same object span a linear subspace with a low dimension. Let us assume, according to the above discussion, that W projects a new input vector into a space orthogonal to the one spanned by the principal components extracted from many views of the object (the "examples"). Then, if the new input is another view of the same object, the result will be close to zero for all units. In the case of the Gaussian, for instance, this means that each unit will be maximally activated and by suitable choice of c any desired output may be synthesized. On the other hand, if the new input is the view of a different object, the result of operating on it with W will be different from zero and possibly large enough to give a very small activity of the unit making it impossible to synthesize a desired output by an appropriate choice of the c (the output will be zero or close to it). In this case, the appropriate W will solve the problem with just one center (since the problem is linear). Notice that if W is symmetric (i.e. if W is the square root of M), it has the same eigenvectors of M , and M and W have the same null space.

6. One may think intuitively that it is desirable that W is space dependent, that is $W = W(x)$. This assumption, however, seems rather meaningless from the point of view of regularization theory. As a consequence, we believe that it is wrong to assume $W = W(x)$ in a scheme such as HyperBF. On the other hand, it makes theoretically sense to use different HyperBF networks for different subsets of the domain of the given multivariate function, each one possibly with a different W . We do not have any theory, however, of how to partition appropriately the domain of the function. An alternative approach, that also makes sense, is local linear approximation. In this case one finds a set of local charts, somewhat similarly to computing $W(x)$.

5.1 A practical algorithm

It seems natural to try to find a reasonable initial value for the parameters c, t_α, M , to start the minimization process. In the absence of more specific prior information the following heuristics seems reasonable.

- Set the number of centers and set the centers' positions to positions suggested by cluster analysis of

the data (or more simply to a subset of the examples' positions).

- Set the rows of W to be vectors orthogonal to the eigenvectors with largest eigenvalues of $\sum_\alpha \sum_i Q_{i,\alpha}$.
- Use matrix pseudo-inversion to find the c_α .
- Use the t_α , $M = W^T W$ and c_α found so far as initial values for gradient descent equations.

It should be noticed that an even more general strategy makes sense in some cases. Suppose that the system can be made to operate satisfactorily with the steps above or perhaps just with the first step. Suppose also that the system can continue to accumulate examples while operating. An example could be an autonomous vehicle that can improve, say, the model of its dynamics by collecting appropriate example pairs while operating. Then it makes sense to perform dimensionality reduction and to move the centers as outlined above. As an additional step one may try to eliminate features that receive little weight, if possible, and then to add other features while keeping the previously found centers. This is equivalent to adding centers of higher dimensionality. Another iteration of moving centers, finding norm weights, eliminating features and centers then takes place.

Experiments with movable centres and movable weights have been performed in the context of object recognition (Poggio and Edelman, 1990; Edelman and Poggio, 1990) and approximation of multivariate functions (Caprile, Girosi and Poggio, 1990) and in both cases the results are promising.

6 Remarks

1. Equation (19) is similar to an operation of (task-dependent) dimensionality reduction (Duda and Hart, 1973) whereas equation (17) is similar to a clustering process.
2. It is conceivable that learning the weights of the norm is even more important than learning the centers and that in many cases it may be preferable to set the centers to a representative subset of the data and to keep them fixed thereafter.
3. A specific matrix W corresponds to a specific metric in the multidimensional input space: W projects the input vector into the subspace spanned by its rows. In the case of the rows of W spanning the space orthogonal to the principal components of the inputs, W assigns a metric ellipsoid with the largest axes (corresponding to a large σ in the Gaussian) along the principal components and the small axis (corresponding to a small σ in the Gaussian) orthogonal to it: thus even vectors that are far away (in the ordinary euclidean metric) are close in this metric if they lie in the hyperplane of the principal components and even close vectors (in the ordinary metric) are far away in the metric induced by W if they are orthogonal to the principal components.
4. In the case of N examples, $n = N$ fixed centers and $M = I$, there are enough data to constrain the

N coefficients c_α to be found. Moving centers add another nd parameters (d is the number of input components) and the matrix M another $\frac{d^2+d}{2}$ independent parameters. Thus the number of examples N must be sufficiently large to constrain adequately the free parameters — n d -dimensional centers, n coefficients c_α and $\frac{d^2+d}{2}$ independent entries of the matrix M . Thus

$$N \gg n + nd + \frac{d^2 + d}{2}.$$

5. In the case of Gaussian basis functions, learning the entries of a diagonal W is equivalent to learning the variances of each two-dimensional (or one-dimensional) Gaussian receptive field for each center. It is clear that sets of units with different scales (see section 3.1) correspond to sets of units with different W .

Acknowledgements We thank Shimon Ullman for useful discussions with one of us (T.P.) that led to an understanding of the role of $\sum_{i=1}^N Q_{i,\alpha}$ in the object recognition example. Lew Tucker and Barbara Moore had suggested weights similar to W long before T.P. finally understood what they meant. We also thanks B. Caprile, C. Furlanello and E. Grimson for useful suggestions and discussions.

References

- [1] A. Albert. *Regression and the Moore-Penrose Pseudoinverse*. Academic Press, New York, 1972.
- [2] R. Basri and S. Ullman. Recognition by linear combinations of models. A.I. Memo No. 1152, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.
- [3] M. Bertero. Regularization methods for linear inverse problems. In C. G. Talenti, editor, *Inverse Problems*. Springer-Verlag, Berlin, 1986.
- [4] D.S. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1988.
- [5] B. Caprile, F. Girosi, and T. Poggio. Hyperbf networks: techniques and experiments. A.I. Memo (in preparation), Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.
- [6] J. Demmel. The geometry of ill-conditioning. *J. Complexity*, 3:201–229, 1987.
- [7] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
- [8] S. Edelman and T. Poggio. Bringing the grandmother back into the picture: a memory-based view of object recognition. A.I. Memo (to appear), Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.
- [9] F. Girosi and T. Poggio. Networks and the best approximation property. A.I. Memo 1164, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.
- [10] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:219–227, 1983.
- [11] J. MacQueen. Some methods of classification and analysis of multivariate observations. In L.M. LeCam and J. Neyman, editors, *Proc. 5th Berkeley Symposium on Math., Stat., and Prob.*, page 281. U. California Press, Berkeley, CA, 1967.
- [12] S.G. Mikhlin. *The problem of the minimum of a quadratic functional*. Holden-Day, San Francisco, CA, 1965.
- [13] J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989.
- [14] V.A. Morozov. *Methods for solving incorrectly posed problems*. Springer-Verlag, Berlin, 1984.
- [15] T. Poggio and S. Edelman. A network that learns to recognize 3D objects. *Nature*, 343:263–266, 1990.
- [16] T. Poggio and F. Girosi. A theory of networks for approximation and learning. A.I. Memo No. 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.
- [17] T. Poggio and F. Girosi. A theory of networks for learning. *Science*, 247:978–982, 1990.
- [18] T. Poggio and F. Girosi. Hyperbf: A powerful approximation technique for learning. In Patrick H. Winston and Sarah A. Shellard, editors, *Artificial Intelligence at MIT: Expanding Frontiers, Vol. 1*. M.I.T. Press, Cambridge, MA, 1990a.
- [19] I. Stakgold. *Green's functions and boundary problems*. John Wiley and Sons, New York, 1979.
- [20] A. N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Math. Dokl.*, 4:1035–1038, 1963.
- [21] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. W. H. Winston, Washington, D.C., 1977.

Model-Group Indexing for Recognition

David Clemens and David Jacobs
MIT Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA 02139

Abstract

In model-based visual recognition, often groups of image features are matched to groups of model features to form initial hypotheses, which are then verified. If all possible matches are considered, this process can require excessive computation. In order to accelerate recognition considerably, the model groups can be arranged in an index space offline (hashed), such that each image group can index into the space and find only those model groups that could have formed that image group. For the case of 3D point model features and 2D point image features, we prove that each model must be represented by a two-dimensional subspace in the index space. We also show that the index space should be of dimension $2G - 4$, where G is the number of features in each group. This places an unexpected lower bound on the space required to implement indexing with 3D models. We discuss the details of how such a space can be theoretically constructed, and introduce an informal method for reasoning about the index spaces based on degrees of freedom. We also discuss practical considerations for implementing the approach, including the significant consequences of image error. We argue that indexing can provide a significant speed-up, particularly as larger groups are formed.

1 Introduction

Computer vision programs that recognize objects often approach the problem by attempting to match features in an image to features in a modeled object. For example, one may match distinctive points in an image, such as corners or extrema of curvature, to comparable model points. Or one may match extended features, such as line segments or curves to sections of a model that will produce such curves. To recognize an object one must find the correct set of such matches. A basic problem with this approach is the exponential number of possible sets of matches. This makes it impractical to explicitly consider every way to match the image features with the model features.

One way of speeding up this matching process is to

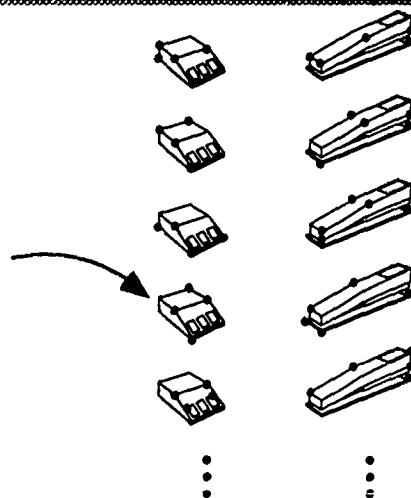
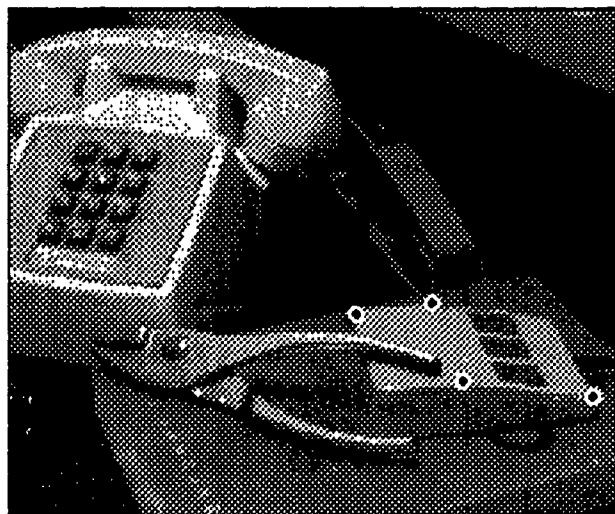


Figure 1: For each group of image features, model-group indexing allows the feasible model groups to be found immediately, without searching through all model groups. Usually, only a small number of model groups are feasible.

use sets of image features to index into a lookup table of model feature sets that has been constructed offline. We will call these sets of features *groups*. At run time, the lookup table will provide the program with only those groups of model features that might feasibly match each group of image features, as depicted in Figure 1. We expect that the number of feasible model groups is much smaller than the total number of model groups. Thus it is useful to avoid the time-consuming process of evaluating and then rejecting infeasible matches. A match is infeasible if there is no possible image of the model group in which its features are aligned with the measured image features, to within error bounds.

In this paper we place a strict, non-trivial lower bound on the amount of space needed to perform indexing of groups of three-dimensional model points using groups of two-dimensional image points. We also show that this bound is tight, by explaining how to construct an indexing system within this bound.

1.1 Related Work

Indexing has been demonstrated in the domain of two-dimensional models, where there is no projection. A system of Kalvin et al.[9] based on more theoretical work by Schwartz and Sharir[13] used indexing to find those two-dimensional model curves that could produce a specific two-dimensional image curve. Wallace[15] similarly performed two-dimensional indexing using a few different kinds of groups, including pairs of vertices connected by a straight line. Jacobs[8] performed grouping of image lines based on proximity and relative orientation, and then used indexing to match these lines to two-dimensional models. Each of these systems had an approach that associated only one or a small number of entries in the indexing lookup table with each group of model features, in the absence of error or occlusion. Accounting for sensing error or partial occlusion of model features may have increased the number of table entries, but still these lookup tables were relatively small and easy to build.

Recently, Lamdan et al.[10][11] have devised a method for building a lookup table for indexing when a two-dimensional model group may be viewed from any three-dimensional viewpoint. That is, the model group consists of a set of coplanar points, but may be oriented three-dimensionally. With their method, a single entry is made in a lookup table for each group of four model points.

It had not been shown whether it was possible to extend Lamdan et al.'s indexing method to apply to three-dimensional models, such that each model may be represented with a single entry in a lookup table. In this paper, we prove that such an extension is not possible. We show that any indexing system for three-dimensional model points and a two-dimensional image must map each set of model points to a two-dimensional surface in an index space. We then show how to construct the smallest such two-dimensional surface analytically. We also discuss many of the practical problems involved in building an indexing system based on these ideas, although some of these problems remain unsolved. In par-

ticular, determining an optimal method of describing the effects of sensing error on such a lookup table has not been addressed. However, although this paper does not describe a practical implementation of an indexing system for three-dimensional objects, we demonstrate some strong restrictions that must apply to any such system.

Before demonstrating these results, we wish to provide some insight about indexing and to describe its advantages. In the next two sections we make the following points:

- With a continuous, error-free image and a continuous index space, for each image group of size four or more, there will only be one model group that could form an image group.
- Considering image error, the number of groups that could cause an image group is proportional to the total number of model groups. Thus, in practice, indexing with groups of size four provides a constant-factor speed-up.
- With larger groups, the speed-up factor increases exponentially with G .
- Indexing is most valuable when coupled with grouping, which can provide large groups of features economically.

2 Background

In the remaining sections of this paper, we work within the following recognition domain. A model of an object consists of a set of 3D features, and an image consists of a set of 2D features. We will discuss point features, although other kinds of features may also be used with indexing. The image is generated by performing a 3D rotation, translation and scaling on one of the models, and projecting it along with various unknown objects into the image plane. Projection is taken to be orthographic (parallel) with scaling (aka "weak perspective"). The goal is to find the correspondence between the image features and the known model features, and to find the pose of the model in the scene.

A common approach to this problem is to consider groups of the image points, and for each image group, to hypothetically match it with groups of the model points. For each match hypothesis, if the groups are large enough, the pose of the model can be determined (or reduced to a small set of possibilities). With the pose known, the hypothetical locations of the other model points in the image can be determined, and further support for the hypothesis can be efficiently discovered (Ayache and Faugeras[1], Lowe[12], Clemens[4], Huttenlocher and Ullman[7]). Such an approach considers $\binom{N}{G}$ image groups, and for each one considers $\binom{M}{G}$ model groups, where N is the number of image points, M is the number of model points in one known object, and G is the size of the group. Its complexity is therefore of order $N^G M^G$. (We exclude the number of operations involved in verifying each hypothesis, since it will be the same for all the methods considered in this paper).

Consider groups of three points. It is well known (see for example Fischler and Bolles[6] or Huttenlocher and

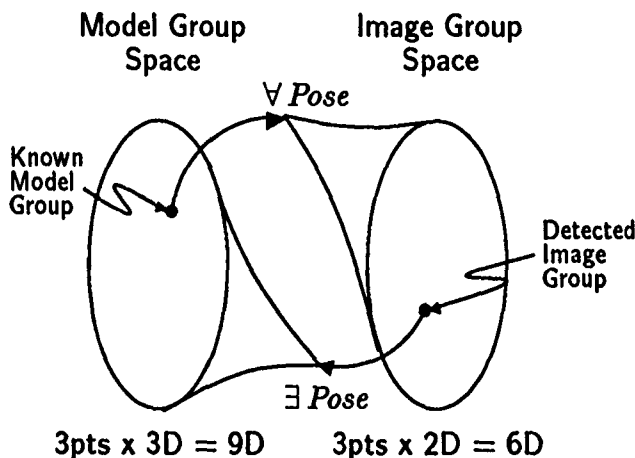


Figure 2: With groups of three points, any model group can correspond to any image group. \forall and \exists are used loosely: for each model group, over all poses, it can form all image groups. Conversely, for each image group, for any model group, there exists a pose such that the model group could have formed the image group.

Ullman[7]) that this is the minimum number of points needed in a match to determine the hypothetical pose (to within a finite number of possibilities). The following equation-counting argument supports this claim: The pose has six degrees of freedom (DOF): two rotation angles to specify the viewing direction, rotation of the image plane about the viewing direction, two translations in the image plane, and scaling. Each point in the image has two degrees of freedom, x and y , which may be used to constrain the pose. (These can be called "degrees of constraint", or DOC). Therefore, it should take at least three image points to fully constrain the pose. Alternatively, the pose has six variables, and each image point provides two equations. If the equations are independent, then the solution space should have zero degrees of freedom. This does not guarantee a unique solution: it means there might be no solution, or a finite number of solutions. Such arguments rely on properties of the equations which may be hard to ascertain, such as their independence, the absence of degenerate cases, and the absence of space-filling mappings. However, they can be helpful in developing intuitions about the problem.

It is also the case that for any three image points and any three (non-colinear) model points, there exists a pose such that the image points are formed by the projection of the model points[7]. So, with groups of three points, every image group could have been caused by every model group (figure 2). Therefore, indexing can offer no assistance in this case. All combinations must be checked, and the time of execution is order N^3M^3 .

Now consider groups of four points. Each image group will have 8 DOC. Intuitively, there are now two extra degrees of constraint in each image group. The extra constraint should be available to discriminate among the many model groups, since not all model groups can cause all image groups. Indeed, with continuous error-free images, we argue that an image group is most likely to be

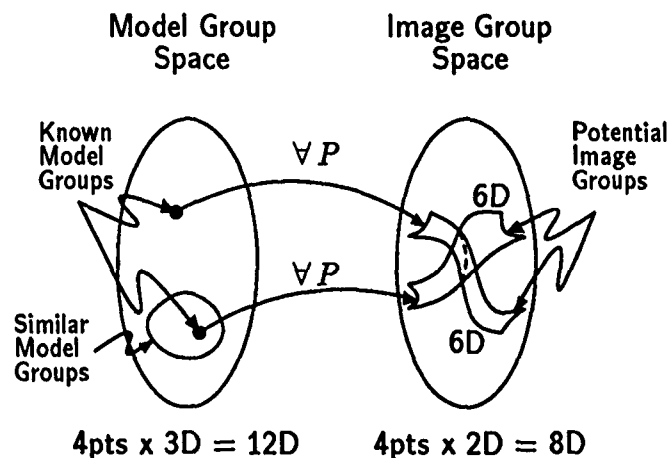


Figure 3: With groups of four points, only some model groups could have formed a particular image group. Each model group can form a 6D subset of the image-group space. Intersections among the potential subsets are of even fewer dimensions.

consistent with only one or zero model groups, depending on whether it was formed from a known model or not, respectively. This implies that the extra constraint in each image group can be used to eliminate all but one of the M^4 model groups as potential matches.

To see why an image group of unknown object features is unlikely to be consistent with any known model groups, consider the space of all possible image groups (Figure 3). With four points in an image group, the space will be 8D. Each point in this space represents a pattern of four points in the image. Each measured image group will correspond to a point in this space. For a particular model group, we will call the set of image groups that it can form its *potential* image groups. A model group will have a potential image group for every pose, since its four points may appear in the image with a different configuration for every pose. Therefore, the subset of the 8D image-group space that contains potential image groups of the model group is at most 6D. With a finite number of known models, there will be a finite number of 6D subsets in the 8D image-group space that correspond to known models—it will be mostly empty. If a point is chosen randomly in the image-group space it is very unlikely to intersect any of the model group potential subsets. (In fact, it will miss them with probability one.) An image group generated from unknown objects is no more likely to hit the potential subsets than a random image group. An image group generated from a known model group *must* lie on that model group's potential subset, but it is just as unlikely to hit a different subset as any other image group.

Alternatively, a geometric intuition may be developed. Consider a particular image group (four points in the image plane) and a particular model group (four points in 3D). We know that we may match any three of the four points in each group, and from that match only two poses of the model group will be possible. With the pose fixed, in order for the fourth model point to appear

at exactly at the four image point, it must lie along a particular line in 3D. Out of all the locations in 3D that it might have, the chances of it lying on the line are very small. However, if the image group was formed from the model group (and the three points were correctly matched), then the fourth point must lie on the line, for one of the two poses.

An exception to this rule occurs if two model groups are similar. That is, there exists a 3D rotation, 3D translation, and scaling that will make them identical. In this case, they will produce exactly the same potential image group subset, and can be dealt with specially.

Therefore, if only the feasible model groups were considered, the run time would be reduced from M^4N^4 to N^4 . This is a tantalizing possibility. However, it was argued for the case of an error-free spatially continuous image. Because of image error and the discretization of the image and the image-group space, each model group must be represented in a small but 8D volume of space, where it would theoretically be represented only in a 6D subset. The potential subset for each model group is still much smaller than the entire 8D image-group space, but the image-group space is no longer sparsely filled, and theoretical speed-up is not attained. However, as the size of a group increases, the image-group space again becomes relatively sparse, and the theoretical result is approached. This important consideration is discussed in greater detail in the section on Considerations for Implementation. For now, we continue with error-free spatially continuous images.

Model-group indexing is a way of using these powerful extra degrees of constraint. Before recognition, markers for the model groups are arranged in an index space. During recognition, the parameters of each image group are used to index into this space, where those models that could possibly form the image are marked. The only requirement is that, for a given image group, the model groups found through indexing are exactly those models which could produce the image group: no more and no fewer. (Indexing would only be less efficient if there were more, but would not work reliably if there were fewer). It is acceptable for several model groups to mark the same point in the index space, and it is acceptable for several image groups to index to the same point in the index space, as long as the stated requirement is met. It is also acceptable for a model group to mark more than one point in index space, or even for an image group to reference more than one point, though these increase storage requirements and reduce the runtime efficiency, respectively.

From the earlier discussion, we can see that such a space exists: it is the image-group space, filled with the model group potential subsets. This could theoretically be formed as follows. for each model group, for all poses, project the model to determine which image groups it can cause. For each of these image groups, place a marker for the model group at that point in the image group index space. At recognition time, each image group finds its point in the image-group space and verifies matches with the model groups marked there, if any.

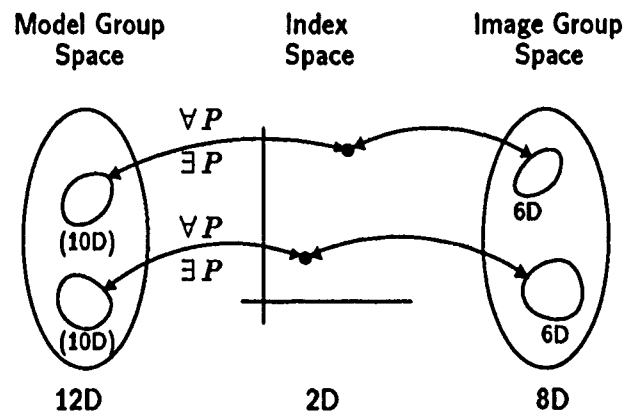


Figure 4: We wish that the index space only needed to represent those dimensions that are available to discriminate among models. We prove that such a space is not possible for general 3D models of any number of points. With coplanar 3D point models, however, the more compact index space is achievable, as demonstrated by Lamdan et al. [10].

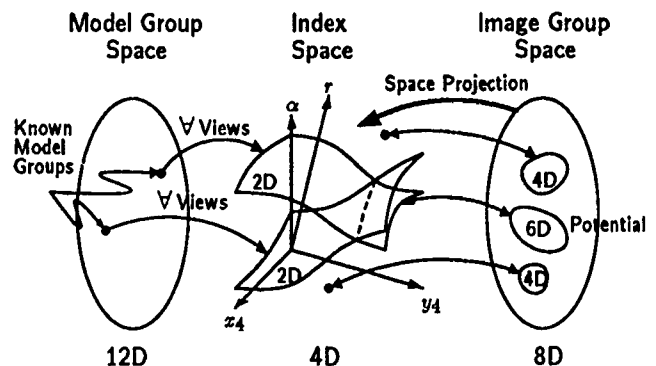


Figure 5: For model groups of four 3D points, allowing only one reference per image group, the smallest index space that uses all the available constraint is 4D, with each model represented along a 2D sheet. The 6D set of potential image groups will reference the sheet in index space. In general, there will be a 4D subset of image groups that all reference the same point in index space.

However, such a space would be impractical to implement, for many reasons. To implement any indexing scheme, we expect to discretize and bound the index space. Because of the high dimensionality of the image-group space, it would require excessive storage. Furthermore, for each model group, the potential image groups must be formed by sampling all six DOF of the pose.

Instead, ideally, the index space would only have to represent the two extra degrees of constraint that are available for model discrimination. Then the index space could be 2D instead of 8D, and each model group would correspond to a point (Figure 4). Lamdan et al.[10] describe such an indexing scheme, but for the special case of planar models. In that case, the fourth point can be represented by two values in terms of the first three points, such that the values are preserved under pose transformation and projection. The two values can then be used for indexing. The main result of this paper is a formal proof that, unfortunately, this is not possible for general 3D point models—the dimensions of the index space cannot be reduced that far.

It may be helpful to note that the reduction of the index space is a *projection* of the image-group space onto a subspace of lower dimension. In this case, projection is meant in the normal mathematical sense, as distinguished from the weak-perspective projection used to form a 2D image from 3D objects. First, the image-group space, along with the potential subsets in it, are remapped to a space of the same dimensionality but with different axes. This is equivalent to changing the representation of the image-group points. Then, the space (and the potential subsets) are projected over some of these axes, call them a_1, a_2 , etc., onto the rest, b_1, b_2 , etc. A point in the projected space has a value for each of the b axes. It corresponds to all the points in the original space that have those same b values, and all other values for the a axes. To project the potential subsets means the following: for each point in the projected space, a model group will be represented if it has a potential image group anywhere in the corresponding part of the image-group space.

In order for an index space to work in general, the projection must be such that the different potential image group subsets would still be distinct in the projected subspace, for every possible set of non-similar model groups.

We prove that each model group must be represented everywhere along a 2D surface in the index space, given that we require each image group to index only a single point in the index space at recognition time. This implies that the most compact projection of the image-group space that preserves model group discrimination is 4D (figure 5). In later sections we describe a specific theoretical 4D index space, and then discuss some important considerations relevant to implementation.

We should point out that indexing need not take the form we give it. For example, Lamdan and Wolfson[11] describe an approach in which each image group indexes a line in index space, instead of a point. However, this increases run time. Furthermore, they use groups of five points, but only a 3D index space. By the reasoning we have presented, we can see that this fails to make use of

two of the degrees of constraint that are available, given that five-point groups are found.

3 Further Motivation for Indexing

Despite the requirement of a 4D indexing space, the approach still offers a theoretical reduction in execution time from order M^3N^3V to order N^4V . The elimination of a dependence on the number of model groups is particularly attractive if large libraries of models are desired, when the total number of model groups can be much greater than the number of image groups. Unfortunately, as mentioned above, because the image cannot be expected to be error-free, the dependence on M returns in full force. To account for error in the image, either an 8D volume of points in the index space must be referenced during recognition, or an 8D volume of possible image groups must be filled by each model group before recognition. Under these circumstances, the number of model groups indexed by an image group is no longer zero or one, but is proportional to the total number of model groups in the space. The entire M^4 factor returns in the order of growth of the run time, but a large constant factor speed-up is retained, which is of practical interest. As discussed later in the implementation section, the size of the speed-up will depend on the size of the image error, and will need to be determined empirically by further research.

Upon entering the practical domain, it is only fair to reconsider the practicality of any method that uses groups of three points. With only three points, any estimate of the model pose in the image will vary considerably due to image error, and may prevent successful verification and recognition. If the group size is increased to four to reduce error in the pose estimate, then execution time will be order M^4N^4 anyway, whether or not indexing is used. In that case, indexing can be employed to achieve a significant constant factor speed-up.

A separate but related motivation for indexing comes from the middle stages of the vision process. All of the above analyses of execution times were based on trying all possible permutations of matches between G model and G image features (where G was four). This includes sets of features from widely different parts of the image. Instead, it should be possible to interpret the image to some extent, and to decide which combinations of image features are more likely to come from a single object, without basing the interpretation on any specific model match. Such interpretation is called *grouping*, and is performed in various ways by many other recognition systems, often without explicit acknowledgment. For example, Acronym[3] grouped image edges into ribbons or ellipses which it matched to portions of generalized cylinders. Bolles and Cain's[2] system grouped together features based on proximity. Lowe[12] first explicitly discussed the importance of grouping to recognition in his very novel and influential system SCERPO. SCERPO grouped together nearby edges based on parallelism, co-termination, and symmetry. These groups of two-dimensional image edges need be matched only to three-dimensional model edges with the same qualities.

The most significant advantages of grouping are at-

tained if the most likely groups can be found without even generating the less likely groups, potentially reducing the N^G factor considerably. In an extreme example, it might be possible to find order N groups in an image. This can be imagined for a grouping algorithm that was based on proximity, for example. Note that model features may also be grouped, based on the image groups they may form (see Lowe[12], for example). Thus, grouping may reduce the search time from order $N^G M^G$ to order NM . Of course, since so few groups are tried, it must be argued that at least one image group from the known object will be found, and that it will be successfully verified despite image error. Nonetheless, any kind of grouping that includes even a single correct image group with high probability can cut the search time drastically by avoiding combinatoric explosion.

Grouping can be expected to aid almost any recognition approach, but it is a particularly good partner for indexing. Indexing needs the larger groups which grouping provides at a reasonable cost, and conversely, indexing is ideal for capitalizing on the increased size of the groups. This is due to two effects. First, the advantages of each method are not reduced by the presence of the other—the speed-up is fully the product of each separate speed-up. Second, with both methods in place, the efficiency may be significantly increased by forming larger groups, while the efficiency of either method alone is likely to decrease with larger groups.

The real power of indexing becomes available as larger groups are used. If indexing with four-point groups causes a speed-up of k , then indexing with larger groups can be expected to cause a substantially larger speed-up: k^{G-3} . (An argument for this order of growth is presented after the index space is described in detail.) In fact, if G is large enough with respect to M , the density of the index space may be reduced so far that the theoretical limit is attained, in which there is no search for each image group. However, without grouping, the number of image groups to consider would rise exponentially with G . The total execution time for recognition with indexing alone would be order $N^G M^G / k^{G-3}$: for each increase of one in G , the time is changed by a factor of NM/k , which is likely to be an increase. On the other hand, recognition with grouping alone is of order $P(G)Q(G)$, where the number of image groups is $P(G)$ and the number of model groups is $Q(G)$. Both of these functions tend to increase with G (but less than exponentially). This is because larger groups are harder to form reliably, and there are more possible combinations of larger groups of a given number of image features. Overlapping groups will almost certainly be required to achieve sufficient reliability. Therefore, with indexing alone or grouping alone, larger groups are not particularly desirable.

When combined, however, execution is of order $P(G)Q(G)/k^{G-3}$. Since P and Q rise much less than exponentially with G , increasing G will decrease the total execution time, possibly by quite a lot. Together, grouping and indexing not only provide their separate advantages, but also support larger groups as a new and powerful tool for decreasing search.

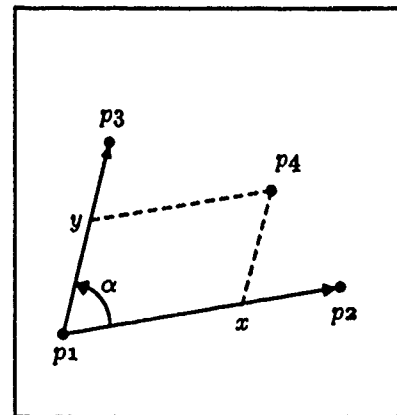


Figure 6: The relative representation of four points in a plane is independent of how the points are translated, rotated, and scaled as a whole. x and y are the relative coordinates of the fourth point—the coordinates in the basis formed by the other three points. α and r describe the basis. $r = \frac{\|p_3 - p_1\|}{\|p_2 - p_1\|}$. The points may be image points or coplanar model points.

In the next section, we outline the proof regarding the lower bound on the size of the indexing space. In the following sections, we describe the method for constructing a theoretical indexing space, and then practical considerations for implementing an indexing scheme for a recognition system.

4 Proof Outline

In the memo version of this paper[5] we present a proof of the following statement: it is not possible to perform model indexing by making only a single entry in an index space for each model group and referencing a single point in index space for each image group, such that exactly those model groups that could have produced the image group are found. In fact, we show that there is a one-to-one mapping from the points in the plane to the entries we must make in index space for each model group. The proof uses a general formulation of index space, so that the result does not depend on any specific choice of representation for this space. This tells us, for example, that Lamdan et al.'s approach cannot be extended to three-dimensional models, while still making only a single entry in index space for each model group. This result, combined with our degrees of constraint arguments made earlier, suggests that the best indexing scheme for groups of four points will map each model group into a two-dimensional surface in a four-dimensional lookup table.

The proof is based on two Lemmas, which we also present here without proof.

Lemma 1: Lamdan et al. point out that if we use three model points as a basis and then represent the remaining points with coordinates relative to that basis, then the relative coordinates will be invariant under projection (Figure 6).

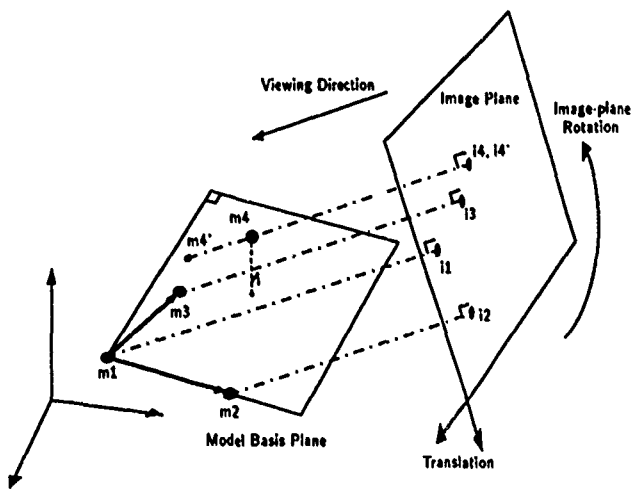


Figure 7: The image points i_1 , i_2 , i_3 , and i_4 are the projections of the model points m_1 , m_2 , m_3 , and m_4 . The values of the image points depend on the pose of the model relative to the image plane. The pose consists of viewpoint (2 DOF), image-plane rotation (1 DOF), image-plane translation (2 DOF), and scale (1 DOF, not shown). In the viewing direction shown, m_4' and m_4 project to the same image point. Note that i_4 has the same relative coordinates as m_4' .

It is this invariance that allows Lamdan et al. to perform indexing by making a single entry in the index space for each model, in the special case where all model points are coplanar. For every ordered set of four model points, they find the relative coordinates of the fourth point, and make an entry at those coordinates in a planar index space.

Lemma 2: Given any non-coplanar model group, and any relative coordinates, (x_{4b}, y_{4b}) , there is always a viewing direction for which i_4 (the image of the fourth model point) has coordinates (x_{4b}, y_{4b}) relative to the first three image points.

We now outline the proof without describing the constructions involved. Suppose M is any model group. We can construct two model groups, $M_{(1,1)}$ and $M_{(2,2)}$, that have some special properties. In each, the fourth point is coplanar with the first three, and, respectively, has the relative coordinates (1,1) and (2,2). There are image groups, $I_{(1,1)}$ and $I_{(2,2)}$, such that $M_{(1,1)}$ can produce $I_{(1,1)}$, $M_{(2,2)}$ can produce $I_{(2,2)}$. Lemma 2 tells us that M can produce both image groups. That means that M must have an entry in index space in common with $M_{(1,1)}$, and an entry in common with $M_{(2,2)}$. However, Lemma 1 can be used to derive that the constructed model groups $M_{(1,1)}$ and $M_{(2,2)}$ can never produce the same image group. This means that they must have no entries in common in index space. Hence, M must make at least two different entries in index space. Finally, we show that we can generate a different model group, $M_{(i,j)}$ for every point in the plane. For each such model group, there will be a different point in the index space at which M must make an entry. Figure 8 depicts this

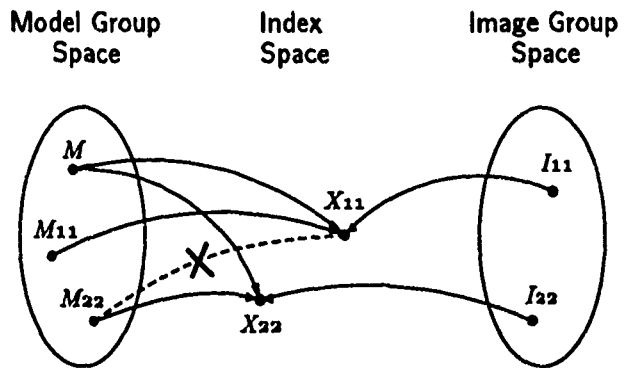


Figure 8: A model, M , can produce images $I_{1,1}$ and $I_{2,2}$, so it must make entries in the index space at the points where these two images look for matches, called $X_{1,1}$ and $X_{2,2}$. Model $M_{2,2}$ can also produce image $I_{2,2}$, so it also has an entry in index space at $X_{2,2}$. Since $M_{2,2}$ could not produce $I_{1,1}$, it may not have an entry at $X_{1,1}$. Therefore, $X_{1,1} \neq X_{2,2}$, and M must make more than one entry in index space.

situation.

This fact tells us to expect each model to map to at least a two-dimensional surface in index space. Our previously developed understanding of degrees of constraint shows that an image of four points contains eight degrees of constraint, two more than the number needed to solve for a model's pose. These two extra DOC allowed Lamdan et al. to map models to points in a two dimensional index space. Similarly, if models must map to a two-dimensional surface, one would expect that surface to be in a four-dimensional index space. That way, the two extra DOC appear as an indexing space that has two degrees of freedom more than the surfaces mapped into it.

5 Theoretical Index Space Construction

We now know that, at the least, we needed to form a two-dimensional surface in index space for every model. In this section we show that this lower bound is in fact a tight bound, by constructing one such conceptual surface. We begin with models that have only four points, and then extend the result to larger models.

From earlier discussion, we see that the index space will be a 4D projection of the 8D image-group space. In doing this, we project the image-group space over the four dimensions that are independent of the index space. But what are the axes of the index space? The potential image groups for each model group appear as 2D sheets in it, parameterized by viewing direction. Viewing direction was defined to be along the line between m_4 and m_4' , where the coordinates of m_4' were (x_{4b}, y_{4b}) . For each model group, there is a point in index space for every (x_{4b}, y_{4b}) . So we will choose x_{4b} and y_{4b} as two of the axes. We will choose the other two axes of the index space so that they also depend only on the viewing direction. Then the projection will be over the remaining DOF of the pose: image-plane rotation, translation, and scale. Fortunately, these four pose DOF correspond

directly to rotation, translation, and scale of the image points. Therefore, in order to implement indexing, we must show the existence of a representation of image groups that has four independent parameters that are also independent of image-plane rotation, translation, and scale. Such representations have already been found, for example, in Thompson and Mundy[14] and Lamdan et al.[10].

We will represent an ordered group of four image points, i_1, i_2, i_3, i_4 , with four parameters: x, y, α , and r . As in the previous section, we will choose three of the points, i_1, i_2 , and i_3 , to use as a basis. x and y will stand for the relative coordinates of i_4 using this basis (just as x_{4b} and y_{4b} did for m'_4). Then, let α be the angle between the line segments $\overline{i_3 i_1}$ and $\overline{i_1 i_2}$, and let r be $\frac{\|\overline{i_3 i_1}\|}{\|\overline{i_1 i_2}\|}$, the ratio of the length of the two line segments. Notice that scaling and translation do not affect these parameters. Furthermore, rotation of a model about an axis normal to the image plane produces rotation of the model's image, which also does not affect these parameters. So in considering all the images a model may produce, we must only be concerned with the effect of rotations about two other axes (viewing direction) on these four parameters.

It is also important to notice that this representation does not discard any information about the model that produced the image. Equivalently, we must show that the representation, when combined with image-plane rotation, translation, and scale, fully specifies the locations of the image points. Image plane translation fully specifies i_1 . The point i_2 is fully described by rotation about i_1 , and the distance between i_1 and i_2 , which is set by scale. i_3 is set by α and r . Finally, i_4 is fully constrained by x and y .

The previous section showed that given four non-coplanar model points, for any (x, y) there exists a pose of the model that will produce an image in which the fourth point, i_4 , has the relative coordinates (x, y) . The point m'_4 was constructed to have relative coordinates (x, y) in the plane formed by the model points m_1, m_2 , and m_3 . i_4 then has relative coordinates (x, y) exactly when we project the model parallel to the line $\overline{m_4 m'_4}$. There are two distinct such projections, depending on whether we view the model from above or below. Viewing the model from below is just like viewing the model from above, except that the coordinate frame is flipped over. Flipping the coordinate frame will not affect the parameters x, y , and r , but it will change α into $2\pi - \alpha$. So, we can map each model into a four-dimensional index space by finding a single value of r and two symmetric values of α for every (x, y) pair. In fact, because the two values are symmetric we may make an entry in index space only at the value $\min(\alpha, 2\pi - \alpha)$, and then calculate the appropriate index value at lookup time.

It is clear that if we project this surface, which is in $x-y-\alpha-r$ space, down into the $x-y$ plane, there is one point on the index surface for every (x, y) pair. Similarly, it is also true that there are two points on the surface for every point in the $\alpha-r$ plane. To see this, we recall that Huttenlocher and Ullman[7] have shown that for any three model points, (m_1, m_2, m_3) , and any three image

points (i_1, i_2, i_3) , there are exactly two poses that will cause (m_1, m_2, m_3) to project to the points (i_1, i_2, i_3) . Without loss of generality, suppose i_1 and i_2 are fixed. For a given r and α , we may calculate the location of i_3 . So, given a four point model, r , and α , we may solve for two possible poses, and then determine at most two possible values of the parameters x and y .

If a model group has more than four points, we may map it to a two-dimensional surface in a higher dimensional index space in the same manner as before. Suppose we have an additional model point, m_5 . We may describe the five image points with the same four parameters as above, plus two additional parameters, x_5, y_5 , the relative coordinates of the fifth point using the first three as a basis. We know that for every pair of parameter values x and y we may determine two viewing directions that produce those values of x and y . Just as we use those viewing directions to determine a single value of α and r for a given (x, y) pair, we also find that the two symmetric viewing directions produce a single pair of values for x_5 and y_5 . So, we may map a model into a two-dimensional surface in the six-dimensional space, with axes α, r, x, y, x_5 , and y_5 . The surface will span the range of all $x-y$ values with one point for each (x, y) pair. Since we could have re-ordered the points, exchanging the fourth and fifth points, it follows immediately that this surface will also have exactly one value for every (x_5, y_5) pair.

Similarly, if the model has more than five points, we may add two more parameters to our index space for every additional point.

This approach to building an index space is somewhat related to work of Thompson and Mundy[14], and of Lamdan and Wolfson [11]. Thompson and Mundy have built a system that does not perform indexing as we use the term, but does use a lookup table to find the pose of an object. They describe an image group that has six DOC so that only two of these DOC depend on the viewing direction. They then fill a two-dimensional table that translates the values of these two parameters into the viewing direction. They build the table by viewing the model from sampled points on the viewing hemisphere. So, at runtime they may determine two of the values of the image group, and look in a table to find, for each model group, the viewing direction that would cause that model group to create that image group. In addition to their indexing work, Lamdan and Wolfson have built a similar system.

This work differs from ours in that it does not use the extra DOC of an image group to discriminate between different model groups that might match it. It is similar, in that it builds a lookup table by sampling the viewing direction. In effect, our approach also determines the appearance of a model from each viewing direction. The approaches are also similar in that they reduce the dimensionality of the lookup table by choosing an image representation in which some parameters vary only with viewing direction, whereas others vary only with image-plane rotation, translation, and scale.

6 Considerations for Implementation

In order to convert the conceptual index space into a practical module for recognition, many issues must be addressed. The index space will be represented as a discrete lookup table. The image is not continuous, but is made up of finite pixels. However, probably the most important issue is error in the localization of the image features. Because of image error, the 6D subset of potential image groups for a given model group will thicken, and become an 8D "plate" in the 8D image group space. This will cause the number of model groups in each bucket of the lookup table to be proportional to the total number of model groups, regardless of how small we make the buckets in the lookup table. The value of the constant of proportionality will depend on how far the image error extends in index space. The shape of the error in the image group space is not difficult to estimate. However, the conversion of the image group parameters into the parameterized representation makes the error difficult to characterize. In addition, the projection of the error into index space is problematic.

In the image-group space, the axes may be considered to be the x_i and y_i of each image point. The error is commonly modeled as independent for each image point, and is bounded by a constant number of pixels. The effect of independent errors in the image points on α , r , x , and y will depend greatly on the pose of the model. For example, when the model appears at a small scale, or the basis points are nearly colinear in the image, then small error in the image points may result in huge variations in the parameters that describe the image group. In that case, the error bounds around the sheet in the index space should be very large. When the image group is large and the basis is stable, the index parameters may be relatively impervious to the same image pixel error, so small error bounds would be appropriate in the index space. However, when we project the image group error "plates" into the index space, we combine the error over all scales. Even though the shape of the 2D sheet is independent of the pose parameters we project over, the error is not.

Therefore, to simply thicken each model's 2D sheet uniformly might not be a reasonable approximation. In order to cover the largest errors, the uniform thickening might have to be so large that a significant fraction of the index space is filled. This would reduce the power of indexing. Alternatively, the model groups could remain thin, but each image group could index a "cloud" of points in the index space at recognition time. This would allow more adaptive error estimation, because the scale is known from the image points. However, it would increase recognition time instead of preparation time. A compromise solution would be as follows: as the model groups fill the index space, they include a coarse encoding of the maximum scale value that would cause that bucket to be included in the error cloud. (In a way, this is like adding an extra axis to index space). At recognition time, for each image group, the scale is estimated and only one bucket is accessed. Of the many model groups found there, only those that have a scale greater than the estimated scale need be considered.

In addition to image error, another difficulty is determining which buckets to fill in the index space for each model group. It might seem reasonable to sample the viewing hemisphere at regular intervals. In practice, this method misses many locations in the index space, because the indices may change very rapidly with viewpoint. In the index space described in this paper, the model's surface is shown to be single-valued in two of the index space parameters. It is easier to determine how quickly the surface is changing, since only the other two parameters need be sampled. This should make preparation of the index space more reliable and efficient.

The theoretical index space has infinite extent. The useful extent of the image group parameters must be determined if the index space is to be represented by a finite array. Bounds should exist based on the facts that pixels are of finite size, and that objects can never cover more than the entire image. It will also probably be appropriate to remap x , y , and r so that the space is more uniformly used, perhaps with logarithms. Even so, the array might be too large to store, especially if larger groups are used. In that case, indexing may be performed for a large group by indexing subsets of the image group, and intersecting the resulting sets of model groups. Again, this introduces a runtime overhead. Another alternative is to hash the higher dimensions, since they will be sparser.

6.1 The Speed-Up Factor

With these considerations in mind, we may explore the practical speed-up to be gained by model-group indexing. The index space will be a finite array of some sort. The indices of the array are (some versions of) x , y , α , and r . Each model group is represented at every (x, y) , by a thickened version of one point in the α - r subspace. Since every model group appears at every (x, y) , we may investigate the density of the entire space by looking at the typical density at each (x, y) . Let e be the average number of buckets each model must fill in the slice of α - r at each (x, y) in order to account for error. Let b be the total number of buckets in each slice of α - r subspace. Then e out of b buckets will be filled by each model at each (x, y) . If Q is the number of model groups, then the typical cell in the index space array will contain pointers to Qe/b model groups. Compared to searching through all Q model groups, this is a speed-up of $k = e/b$.

We may now speculate on the benefit of larger groups. With five points, two more axes will be added to the index space. It is reasonable to assume that each model group will need to fill e buckets out of b in each of the new dimensions, for a total of e^2 out of b^2 at each (x, y) . Thus, the speed-up will be $e^2/b^2 = k^2$. In general, we expect the speed-up to be k^{G-3} . The value of k will need to be determined empirically, based on the image-error due to the feature detector, and on the shape of this error in the index space.

6.2 Other Practical Advantages

In addition to this speed-up, the index space has several other practical advantages for recognition. For each match found between an image group and a model group,

it is usually necessary to solve for the pose of the model in order to verify the match. Along with pointers to the model group, the index space may be filled with good pose estimates that aid the pose solution. This can be done because the viewpoint is known when the index space is filled. That is, we may include the advantages of the lookup table used by Thompson and Mundy. Also, views of model groups that are not visible due to self-occlusion may be omitted. This includes any view that is unlikely to cause a detectable image group, depending on the way image groups are formed.

Also note that representation of the image points depends on choosing one as i_1 , another as i_2 , etc. The roles are not symmetrical. If the assignment is chosen differently, the index parameters will change. There needs to be a canonical way of choosing the order of the points, such as forming the convex hull and starting with the longest distance between hull points, and continuing through angles to each of the other points in order. When the order is ambiguous, both possibilities must be tried. Ordering the points has a practical advantage, however: the order in which they match the model group is known. Without a canonical ordering, all $G!$ matches would have to be tried, whether indexing was being used or not.

7 Conclusion

We have shown that, in theory, an index space can be a powerful tool for reducing the image-model match search from order $N^G M^G$ to order N^G , when $G > 3$. However, if image groups are to index a single point at recognition time, then the index space must contain pointers to each model group over a 2D sheet, and should therefore be 4D. In practice, the presence of image error prevents the N^G result from being attained, but indexing still improves the search by a factor that increases exponentially in G : $N^G M^G / k^G$. When combined with grouping, indexing utilizes the advantages of larger group sizes to provide a match time of order $P(G)Q(G)/k^G$. This, along with other advantages, makes indexing an attractive practical approach worthy of further research.

References

- [1] Ayache, N. and Faugeras, O., 1986. "HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):44-54.
- [2] Bolles, R. and Cain, R., 1982. "Recognizing and Locating Partially Visible Objects: The Local-Feature-Focus Method." *The International Journal of Robotics Research*, 1(3):57-82.
- [3] Brooks, R., 1981. "Symbolic Reasoning Among 3-D Models and 2-D Images." *Artificial Intelligence*, 17:285-348.
- [4] Clemens, D., 1986. The Recognition of Two-Dimensional Modeled Objects in Images, Master's Thesis, MIT Department of Electrical Engineering and Computer Science.
- [5] Clemens, D. and Jacobs, D., 1990. *Model-Group Indexing for Recognition*. MIT AI Memo 1246.
- [6] Fischler, M. and Bolles, R., 1981. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Analysis and Automated Cartography." *Communications of the Association of Computing Machinery*, 24(6):381-395.
- [7] Huttenlocher, D. and Ullman, S., 1989. "Recognizing Solid Objects by Alignment with an Image." Cornell University TR 89-978.
- [8] Jacobs, D., 1988. *The Use of Grouping in Visual Object Recognition*. MIT AI Memo 1177.
- [9] Kalvin, A., Schonberg, E., Schwartz, J., and Sharir, M., 1986. "Two-Dimensional, Model-Based, Boundary Matching Using Footprints." *The International Journal of Robotics Research*, 5(4):38-55.
- [10] Lamdan, Y., Schwartz, J. and Wolfson, H., 1988. "Object Recognition by Affine Invariant Matching." *Proceedings on Computer Vision and Pattern Recognition*:335-344.
- [11] Lamdan, Y. and Wolfson, H., 1988. "Geometric Hashing: A General and Efficient Model-Based Recognition Scheme." *Proceedings of the IEEE Conference on Robotics and Automation*:238-249.
- [12] Lowe, D. 1985. *Perceptual Organization and Visual Recognition*. The Netherlands: Kluwer Academic Publishers.
- [13] Schwartz, J. and Sharir, M., 1987. "Identification of Partially Obscured Objects in Two and Three Dimensions by Matching Noisy Characteristic Curves." *The International Journal of Robotics Research*, 6(2):29-44.
- [14] Thompson, D. and Mundy, J. 1987. "Three-Dimensional Model Matching from an Unconstrained Viewpoint." *Proceedings of the IEEE Conference on Robotics and Automation*:208-220.
- [15] Wallace, A., 1987. "Matching Segmented Scenes to Models Using Pairwise Relationships Between Features." *Image and Vision Computing*, 5(2):114-120.

The Skeleton Sketch: Finding Salient Frames of Reference

J. Brian Subirana-Vilanova
MIT Artificial Intelligence Laboratory
Cambridge, MA 02139
email: brian@ai.mit.edu

Abstract

In this paper we present a novel definition of curved axis of inertia and a scheme for finding a frame of reference of a shape in an image based on such a definition. And we discuss how the frame can be used to describe the shape. The scheme assigns a saliency measure to each component of the reference frame that is a measure of its relevance, so that large and central parts play a more central role in the description of the shape. The scheme also computes a major axis that is used to organize the description of the shape, so that a canonical description can be obtained. One of the remarkable features of the scheme is its tolerance to noisy and spurious data. Several perceptual phenomena observed in humans such as grouping based on symmetry and environmental bias in shape description can be reproduced naturally in this scheme. The scheme also supports other operations such as finding the most "interesting" point in the image or defining what is inside and what is outside an object. An extension of the scheme to find high, long and smooth curves on an arbitrary surface is presented. The extension is illustrated on the problem of finding salient blobs in images and it is suggested that similar schemes be used in other early and middle level vision tasks.

1 Introduction

A shape description is an encoding of a shape. A common approach is to describe the points of the shape in a cartesian coordinate reference frame fixed in the image (fig. 1). An alternative is to center the frame on the shape so that a canonical description can be achieved. For some shapes this can be obtained by orienting the frame of reference along the inertia axis of the shape (see fig. 1). If the objects are elongated and flexible this solution is not appropriate, instead, a curved frame of reference can be used (fig. 3). In this case the frame can also be used to find a canonical description of the shape by "unbending" it using the frame as an anchor structure (fig. 3). In this paper, we address the problem of finding such reference frames and how they can be used to describe shapes.

Finding reference frames is a straightforward problem for simple geometric shapes such as a square or a rect-

angle. The problem becomes difficult for shapes that do not have a clear symmetry axis such as a notched rectangle, (for some more examples see figs. 3 and 11,) and none of the schemes presented previously can handle them successfully. Ultimately, we would like to achieve human like performance. This is difficult partly because what humans consider to be a good skeleton can be influenced by high-level knowledge (see fig. 2). The problem is twofold: establishing a definition for the frame of reference of a shape, and finding an algorithm that computes it.

The study of reference frames has received considerable attention in the computer vision literature. Reference frames have been used for different purposes and given different names. Previous schemes for computing skeletons usually fall into one of two classes. The first class looks for straight axes, such as the axis of inertia. These methods are global (the axis is determined by all the contour points), and they produce a single straight axes. The second class can find a curved axis along the figure, but the computation is based on local information. That is, the axis at a given location is determined by small pieces of contours surrounding this location. Examples of such schemes are, to name but a few, Morphological Filters (see [Serra 82] for an overview), Distance Transforms [Rosenfeld and Pfaltz 68], [Borgefors 86], [Arcelli, Cordella and Levialdi 81], Symmetric Axis Transform [Blum 67], [Blum and Nagel 78] and Smoothed Local Symmetries [Brady and Asada 84], [Connell and Brady 87]. Recently, computations based on physical models have been proposed by [Brady and Scott 88] and [Scott, Turner and Zisserman 89]. In contrast, the novel scheme presented in this paper, which we call Curved Inertia Frames (C.I.F.), can extract curved symmetry axes, and yet use global information.

The organization of the paper is as follows. The approach that we present for finding skeletons is divided into two successive stages. In Section 2, we present the first stage, in which we obtain two local measures at every point: the *inertia value* and the *tolerated length*, which will provide a local symmetry measure at every point, and for every orientation. This measure is high if locally the point in question appears to be a part of a symmetry axis. This simply means that, at the given orientation, the point is equally distant from two image

contours. The symmetry measure therefore produces a map of potential fragments of symmetry curves. In Sections 3 and 4, we present the second stage in which we find long and smooth axes that go through points of high inertia values and tolerated length. In section 5 we introduce the skeleton sketch and we show some results and applications of the scheme, and in section 6 we discuss the relation of our scheme to human perception. We conclude in section 7 by presenting an extension of the scheme to find high, long, and smooth curves on an arbitrary surface. The extension is illustrated on the problem of finding salient blobs in images.

In Appendix I we show that the class of measures that the computation described in sections 3 and 4 can compute is very limited.

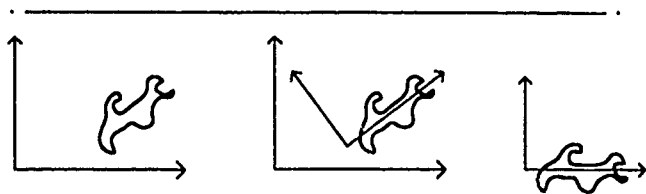


Figure 1: *Left*: a shape described in a image or viewer centered reference frame. *Center*: the same shape with an object centered reference frame superimposed on it. *Right*: a canonical description of the shape.



Figure 2: All the shapes in this fig. have been drawn by adding a small segment to the shape in the middle. At a first glance, all of these shapes would be interpreted as two blobs. But if we are told that they are letters then finer distinctions are made between them. When we use such high level knowledge we perceive these shapes as being different and therefore their associated skeletons would differ dramatically.



Figure 3: Which two of the three shapes on the left are more similar? One way of answering this question is by "unbending" the shapes using their skeleton as a reference frame, which results in the three shapes on the right. Once the shapes have been unbent, it can be concluded using simple matching procedures that two of them have similar "shapes" and that two others have similar length. We suggest that the recognition of elongated flexible objects can be performed in some cases by transforming the shape to a canonical form and that this transformation can be achieved by unbending the shape using its skeleton as an anchor structure. The unbending presented in this fig. was obtained using an implemented lisp program.

2 Inertia Surfaces and Tolerated Length

Previously presented computations to find a curved axis generally suffer from three problems. First, they produce disconnected skeletons for shapes that deviate from a perfect symmetry or that have fragmented boundaries. Second, they are unstable in that the obtained skeleton can change drastically for a small change in the shape (e.g. a notched rectangle vs a rectangle). Third, they do not assign any measure to the different components of the skeleton that indicates the "relative" relevance of the different components of the shape. In addition, many computations depend on scale, and this introduces the problem of determining the correct scale. It is unclear also what to do with shapes that are somewhat circular because they do not have a clear symmetry axis. Heide [84], [Bagley 85], [Brady and Connell 87], [Fleck 86], [Fleck 89] suggest to solve the stability problem by post-processing the SLS, eliminating the portions of it that are due to noise, connecting segments that come from adjacent parts of the shape, and smoothing the contours at different scales. Fleck [86] designed a separate computation to handle circular shapes, the Local Rotational Symmetries.

If we are willing to restrict the frame to a single straight line then the axis of least inertia is a good choice because it provides a connected skeleton and it can handle non symmetric connected shapes. The inertia $\text{In}(SL, A)$ of a shape A with respect to a straight line SL is defined as:

$$\text{In}(SL, A) = \int_A \mathcal{D}(a, SL)^2 da \quad (1)$$

The integral is extended over all the area of the shape, and $\mathcal{D}(a, SL)$ denotes the distance from a point a of the shape to the line SL . The axis of least inertia of a shape A is defined as the straight line SL that minimizes $\text{In}(SL, A)$.

A naive way of extending the definition of axis of least inertia to handle bent curves would be to use eq. 1, so that the skeleton be defined as the curve C that minimizes $\text{In}(C, A)$. This definition is not useful if C can be any arbitrary curve because a highly bent curve that goes through all points inside the shape would have zero inertia (see fig. 4). There are two possible ways to avoid this problem: either we define a new measure that penalizes such curves or we restrict the set of curves that we can use to minimize the inertia. We chose the former approach and we call the new measure defined in this paper the *inertia*, the *skeleton saliency* or *saliency* of the curve. The skeleton saliency of a curve will depend on two local measures: the *inertia value* I that will play a role similar to that of $\mathcal{D}(p, a)$ in eq. 1 and the *tolerated length* T that will prevent non-smooth curves from receiving optimal saliency values. We define the problem as a maximization problem so that the best skeleton will be the curve that has the highest saliency value. The saliency of a curve will be defined in eq. 4 and it is defined for a curve C of length L that starts at a given point p in the image.

The inertia value

The inertia measure \mathcal{I} for a point p and an orientation α is defined as: $\mathcal{I}(p, \alpha) = 2R \frac{(R-r)^s}{R^s}$, where r and R are defined in fig. 5. For a given orientation, the inertia values of the points in the image form a surface that we call the *inertia surface* for that orientation. Fig. 4 illustrates why the inertia values should depend on the orientation of the skeleton and fig. 6 shows the inertia surfaces for a square at four orientations.

Local maxima on the inertia values for one orientation indicate that the point is centered in the shape at that orientation. The absolute value of the local maximum indicates how large the section of the body is at that point for the given orientation, so that points in large sections of the body receive higher inertia values. The constant s or *symmetry constant*, 2 in the actual implementation, controls the decrease in the inertia values for points away from the center of the corresponding section, the larger s is the larger the decrease. If s is very large only center points obtain high values and if $s = 0$ all points of a section receive the same value.

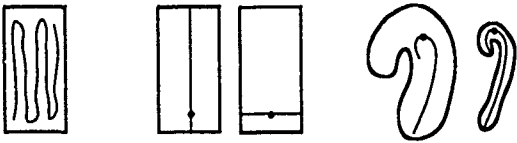


Figure 4: Left: A rectangle and a curve that would receive very low inertia according to eq. 1. Center: Evidence that the inertia value of a point should depend on orientation. Right: Evidence that the tolerated curvature on a skeleton should depend on the width of the shape.

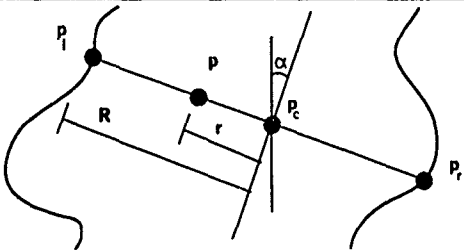


Figure 5: This figure shows how the inertia surfaces are defined for a given orientation α . The value for the surface at a point p is $\mathcal{I}(R, r)$. The function \mathcal{I} or *inertia function* is defined in the text. $R = d(p_l, p_r)/2$ and $r = d(p, p_c)$, where p_l and p_r are the points of the contour that intersect with a straight line perpendicular to α that goes through p at opposite directions and p_c is the midpoint of the interval between these two points. If there is more than one intersection along one direction then we use the nearest one. If there is no intersection at all then we give a preassigned value to the surface, 0 in the current implementation.

The tolerated length

We define the *tolerated length* T for a curvature of radius r_c as 0 if $r_c < R+r$ and $r_c(\pi - \arccos(\frac{r_c - (R+r)}{r_c}))$ otherwise.

Figure 4 provides evidence that the curvature on a skeleton should depend on the width of the shape. The *tolerated length* will be used to evaluate the smoothness of a frame so that the curvature that is *tolerated* depends on the width of the section so that high curvature is only allowed on thin sections of the shape. The saliency of a curve will be the sum of the inertia values "up to" the tolerated length so that for a high tolerated length, i.e. low curvature, the sum will include more terms and will be higher. A curve that bends into itself within a section of the shape will have a point within the curve that will have 0 tolerated length so that the saliency of the curve will not depend on the shape of the curve beyond that point.

In this section we have introduced the inertia surfaces and the tolerated length. We will define a salient frame of reference to be a high and long curve in the inertia surfaces that is as smooth as possible based on the tolerated length. In the next section we will investigate how such a curve might be computed in a general framework and in section 4 we will see how to include the inertia values and the tolerated length in the computation and what is the definition of the saliency measure that results.

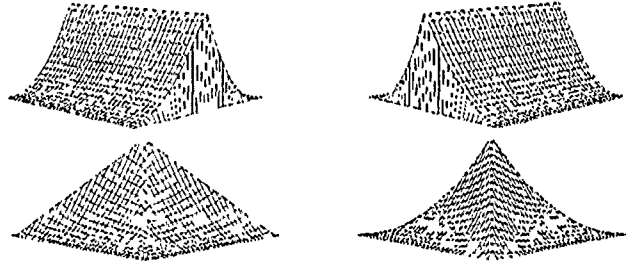


Figure 6: Plots of the inertia surfaces for a square for orientations parallel to the sides (left two plots) and parallel to the diagonals (right two plots).

3 A network to find salient curves

In this section we will derive a class of dynamic programming algorithms that find curves in an arbitrary graph that maximize a certain quantity. In the next section we will apply these algorithms to finding high, long, and smooth curves in the inertia surfaces. [Mahoney 87] showed that long and smooth curves in binary images are salient in human perception even if they have multiple gaps and under the presence of other curves. [Sha'ashua and Ullman 88] devised a saliency measure and a dynamic programming algorithm that can find such salient curves in a binary image. We build on their work and show how their ideas can be extended to deal with arbitrary surfaces. In this section we will examine their computation in a way geared at demonstrating that the kind of saliency measures that can be computed with the network is very limited, the actual proof of this will be given in Appendix I.

We define a *directed graph with properties* $G = (V, E, P_E, P_J)$ as a graph with a set of vertices $V = \{v_i\}$; a set of edges $E = \{e_{i,j} = (v_i, v_j) \mid \text{s.t. } v_i, v_j \in V\}$; a function $P_E : E \rightarrow \mathcal{R}_c$ that assigns a vector p_e of proper-

ties to each edge; and a function $P_J : \mathcal{R}_j \rightarrow \mathcal{R}_j$ that assigns a vector p_j of *properties* to each *junction* where a junction is a pair of adjacent edges. We will refer to a curve in the graph as a sequence of connected edges. We assume that we have a *saliency function* S that associates a positive integer $S(C)$ with each curve C in the graph. This integer is the *saliency* or *saliency value* of the curve. The saliency of a curve will be defined in terms of the properties of the elements (vertices, edges and junctions) of the curve. Our problem is to find a computation that finds for every point and everyone of its connecting edges, the most salient curve starting at that point with that edge. This includes defining a saliency function and a computation that will find the salient curves for that function. The applications that will be shown here work with a 2 dimensional grid. The vertices are the points in the grid and the edges the elements that connect the different points in the grid. The junctions will be used to include in the saliency function *properties* of the shape of the curve such as curvature.

The computation will be performed in a locally connected parallel network with a processor $pe_{i,j}$ for every edge $e_{i,j}$. The processors corresponding to the incoming edges of a given vertex will be connected to those corresponding to the connecting edges at that vertex. We will design the computation so that we know at iteration n what is the saliency of the most salient curve of size n for every edge. This provides a constraint in the invariant of the algorithm that we are seeking that will guide us to the final algorithm. In order for the computation to have some computing power each processor $pe_{i,j}$ must have at least *one* state variable that we will denote as $s_{i,j}$. Since we want to know the saliency of the most salient curve of length n starting with any given edge, we will assume that, at iteration n , $s_{i,j}$ contains that value for that edge. Observe that having only one variable looks like a big restriction, however, we show in Appendix I that allowing more state variables does not add any power to the possible saliency functions that can be computed with this network. Since the saliency of a curve is defined only by the properties of the elements in the curve, it cannot be influenced by properties of elements outside the curve. Therefore the computation to be performed can be expressed as:

$$s_{i,j}(n+1) = \text{MAX}\{\mathcal{F}(n+1, p_e, p_j, s_{i,j}(n), s_{j,k}(n)) \mid (j,k) \in E\}$$

$$s_{i,j}(0) = \mathcal{F}(0, p_e, p_j, 0, 0) \quad (2)$$

Where \mathcal{F} is the function that will be computed in every iteration and that will lead to the computed saliency. Observe that given \mathcal{F} , the saliency value of any curve can be found by applying \mathcal{F} recursively on the elements of the curve.

We are now interested in what type of saliency functions S we can use and what type of functions \mathcal{F} are needed to compute them such that the value that we obtain in the computation is the maximum for the resulting saliency measure S . Using contradiction and induction we conclude that a function \mathcal{F} will compute the

most salient curve for all possible graphs if and only if it is monotonically increasing in its last argument i.e. iff $\forall p, x, y \ x < y \rightarrow \mathcal{F}(p, x) < \mathcal{F}(p, y)$, where p is used to abbreviate the first four arguments of \mathcal{F} .

What type of functions \mathcal{F} verify this condition? We expect them to behave freely as p varies. And when $s_{j,k}$ varies, we expect \mathcal{F} to change in the same direction with an amount that depends on p . A simple way to fulfill this condition is with the following function:

$$\mathcal{F}(p, x) = f(p) + g(x) * h(p) \quad (3)$$

where f, g and h are positive functions and g is monotonically increasing.

We now know what type of function \mathcal{F} we should use but we do not know what type of saliency measures we can compute. Let us start by looking at the saliency S_i that we would compute for a curve of a length i . For simplicity we assume that g is the identity function:

$$S_i = S_{i-1} + f(p_{i,i-1}) * \prod_{k=1}^{i-1} h(p_{k,k+1}) = \sum_{l=1}^i f(p_{l,l-1}) * \prod_{k=1}^{l-1} h(p_{k,k+1}).$$

The value computed is the sum of the $f(p_{i,j})$'s along the curve weighted by the product of the $h(p_{i,j})$'s. Using $0 \leq h \leq 1$ we can ensure that the total saliency will be smaller than the sum of the f 's. One way of achieving this is by using $h = 1/k$ or $h = \exp(-k)$ and restricting k to be larger than 1. The f 's will then be a quantity to be maximized and the k 's a quantity to be minimized along the curve. In the skeleton network presented in the next section, f will be the inertia measure and k will depend on the tolerated length and will account for the shape of the curve so that the saliency of a curve is the sum of the inertia values along a curve weighted by a number that depends on the overall smoothness of the curve.

At step n , the network as designed will know about the most salient curve of length n starting from any edge. Recovering the most salient curve from a given point can be done by tracing the links chosen by the processors (from eq. 2).

4 Finding high, long, and smooth curves

In this section we will show how the network defined in the previous section can be used to find frames of reference using the *inertia surfaces* and the *tolerated length* as defined in Section 2. The *directed graph with properties* that defines the network to be used has one vertex for every pixel in the image and one edge connecting it to each of its neighbors thus yielding a locally connected parallel network. This results in a network that has eight orientations per pixel. The number of orientations per pixel can be increased to improve the accuracy of the output.

The functions f, g and h (see eq. 3) are defined as: $f(p) = f(p_e) = \mathcal{I}(R, r)$, $g(x) = x$ and $h(p) = h(p_j) = \frac{1}{\rho^{\frac{1}{\alpha} \cdot \frac{1}{\sigma} \cdot (p_j)}}$. α , which we call the *circle constant*, scales the tolerated length, and it was set to 4 in the current implementation. ρ , which we call the *penetration factor*,

was set to 0.5. And l_{emt} is the length of the corresponding element. Also, $s_{i,j}(0) = 0$ because the saliency of a skeleton of length 0 should be 0.

With this definition the saliency value assigned to a curve of length L is:

$$S_L = \sum_{l=1}^L I(p_{l,l-1}) \prod_{k=1}^{l-1} \rho^{\frac{l_{emt}}{\alpha T(p_k)}} = \sum_{l=1}^L I(p_{l,l-1}) \rho^{\sum_{k=1}^{l-1} \frac{l_{emt}}{\alpha T(p_k)}}$$

Which is an approximation of the continuous value given in eq. 4 below. Where S_L is the saliency of a parameterized curve $C(u)$, and $I(u)$ and $T(u)$ are the inertia value and the tolerated length respectively at point u of the curve.

$$S_L = \int_0^L I(l) \rho^{\int_0^l \frac{1}{\alpha T(t)} dt} dl \quad (4)$$

The obtained measure favors curves that lie in large and central areas of the shape and that have a low overall internal curvature. The measure is bounded by the area of the shape. A straight symmetry axis of a convex shape will have a saliency equal to the area of the shape. In the next section we will present some results that show the robustness of the scheme in the presence of noisy shapes.

Observe that if the tolerated length $T(t)$ at one point $C(t)$ is small then $\int_0^l \frac{1}{\alpha T(t)} dt$ is large so that $\rho^{\int_0^l \frac{1}{\alpha T(t)} dt} dl$ becomes very small (since $\rho < 1$) and so does the saliency for the curve S_L . A small α or ρ hence penalize curvature favoring smoother curves.

The actual implementation of the network included a smoothing term that enabled the processors to change their orientation at each iteration instead of keeping only one of the eight initial orientations. Since we are searching for smooth curves, the new orientation is computed by looking at the nearby pixels of the curve at each iteration so that the total curvature is minimized.

5 Results and applications

In this section we will present some results and applications of the frame computation and in the next section we will discuss the connections of our findings to human perception.

The network described in the previous section has been implemented on a Connection Machine and tried on a variety of images. The implementation works in two stages. First, the distance to the nearest point of the shape are computed at different orientations all over the image so that the *inertia surfaces* and the *tolerated length* can be computed, this requires a simple distance transform of the image. In the second stage, the network described in section 4 computes the saliency of the best curve starting at each point in the image for different orientations - eight in the current implementation. The number of iterations needed is bounded by the length of the most salient curve but in general a much smaller number of iterations will suffice. In all the examples shown in this paper the images where 128 by 128 pixels and 128 iterations where used. However, in most of the examples, the results do not change after about 40 iterations. In general, the number of iterations needed is bounded by the width of the shape measured in pixels.

The skeleton sketch and the most salient curve:

The *skeleton sketch* contains the saliency value for the most salient curve at each point. The skeleton sketch is similar to the saliency map described in [Sha'ashua and Ullman 88] and [Koch and Ullman 85] because it provides a saliency measure at every point in the image. Fig. 10 shows the skeleton sketch for a square. The best skeleton can be found by tracing the curve that starts at the point that has the highest skeleton saliency value. Fig. 11 shows a few shapes and the most salient curve found by the network for each of them. Observe that the algorithm is very robust in the presence of non smooth contours. Given a region in the image we can find the best curve that starts in the region by finding the maxima of the skeleton sketch in the region, see fig. 11. In general, any local maximum in the skeleton sketch corresponds to a curve that accounts for a symmetry in the image. Local maxima that lie in the shape itself are particularly interesting.

The most salient point:

In many vision tasks, besides being interested in finding a salient skeleton, we are interested in finding a particular point related to the curve, shape or image. This can be due to a variety of reasons, because it defines a point in which to start subsequent processing to the curve or because it defines a particular place in which to shift our window of attention. Different points can be defined, the point with the highest saliency value is one of them.

Another interesting place in the image is the most central point in a curve which can be computed by our scheme by looking for the saliency values along the curve at both directions within the curve. The most central point can be defined as the point where these two values are "large and equal", the point that maximizes $\min(p_l, p_r)$ has been used in the current implementation, other functions are possible, see fig. 11 for some examples. Observe in fig. 11 that a given curve can have several *central points* due to different local maxima.

The most central point in the image can be defined similarly as the point that maximizes $\min(p_l, p_r)$ for all orientations.

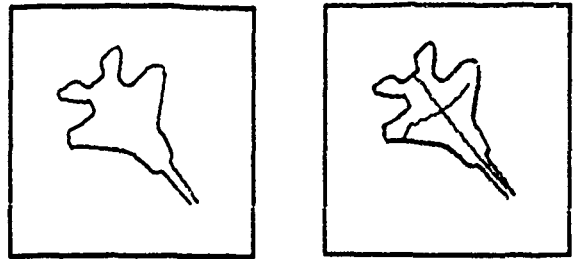


Figure 7: An airplane and its skeleton as found by our scheme.

Shape description:

Each locally salient curve in the image corresponds to a *symmetric region* in one portion of the scene. The se-

lection of the set of most interesting frames corresponding to the different parts of the shape yields a part description of the scene. Doing this is not trivial because a salient curve is surrounded by other curves of similar saliency, in general, a curve displaced one pixel to the side from the most salient curve will have a saliency value similar to that of the most salient one and higher than that of other locally most salient curves. In order to inhibit these curves we color out from a locally maximal curve at perpendicular directions to suppress parallel nearby curves, the amount to color can be determined by the average width of the curve. Once nearby curves have been suppressed we look for the next most salient curve and iterate this process. Fig. 7 shows the skeleton found for an airplane. The skeleton can then be used to find a part description of the shape in which each component of the frame has different elements associated that describe it: a set of contours from the shape, a saliency measure that reflects the relevance or saliency that the component has within the shape, a central point, a location within the shape.

Inside-outside:

The network can also be used to determine a continuous measure of inside-outside. The distance from a point to the frame can be used as a measure of how near the outside of the shape is the point. This measure can be computed using a scheme similar to the one used to inhibit nearby curves as described in the previous paragraph: coloring out from the frame at perpendicular orientations, and using the time where a point is colored as a measure of how far from the frame the point is. The saliency of a curve provides a measure of the area swept by the curve which can be used to scale the coloring process.

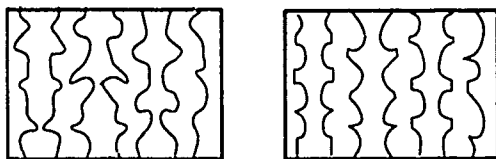


Figure 8: This fig. illustrates the importance of symmetry and convexity in grouping. The curves in the left image are grouped together based on symmetry. On the right image, convexity overrides symmetry, after [Kanizsa and Gerbino 76]. This grouping can be performed with the network presented in this paper by looking for the salient axes in the image.

6 Relation to human perception

The skeleton found by the network for a given shape agree in general with the skeleton that humans would assign to it. In this section we show how the scheme can handle various peculiarities of human perception.

Three frames of reference are important in the perception of shape and spatial relations by humans. that of the perceived object, that of the perceiver and that of the environment. In this paper we have concentrated on the first of them. In some cases the perception of the shape can be biased by the frame of the environment, in

particular humans have a bias for the vertical in shape description so that some shapes are perceived very differently depending on the orientation at which they are viewed, for example a rotated square is perceived as a diamond. This bias can be taken into account in our scheme by adding some constant value to the inertia surface that corresponds to the vertical orientation so that curves that are vertical receive a higher saliency value. Adding the bias towards the vertical is also useful because it can handle non elongated objects that are not symmetric, so that the preferred frame is a vertical axis that goes through the center of the shape. Another alternative is to define a specific computation to handle the portions of the shapes that are circular [Fleck 86], [Brady and Scott 88].

In other cases, the preferred frame is defined by the combination of several otherwise non salient frames. This is the case in Mach's demonstration (see fig. 11), which was first described by E. Mach at the beginning of this century. Our scheme incorporates this behavior because the best curve can be extended beyond one object so that the saliency of one axis is increased by the presence of objects nearby, especially when the objects have salient axis that are aligned. This example also illustrates the tolerance to fragmented shapes that the scheme has.

In figure-ground segregation and grouping it is well known that humans prefer symmetric regions over those that are not (fig. 8). Symmetric regions can be discerned in our scheme by looking for the points in the image with higher skeleton saliency values. [Kanizsa and Gervino 76] have shown that in some cases convexity may override symmetry, see fig. 8. Convexity information can be introduced in the inertia surfaces by looking at the distances to the shape and at the convexity at these points so that frames inside a convex region receive a higher symmetry value. Observe that the relevant scale of the convexity at each point can be determined by the distances to the shape R and r .

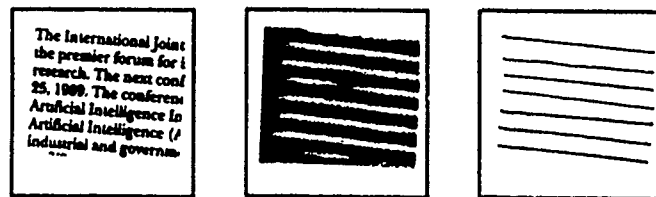


Figure 9: *Left:* Text image. *Center:* Output of the convolution of the text image with an elongated horizontal gabor filter. *Right:* Most salient curves.

7 Conclusion and future research

In this paper we have presented C.I.F. (Curved Inertia Frames), a novel scheme to compute curved symmetry axes. Previous schemes either use global information but compute only straight axis or compute curved axis and use only local information. The scheme presented in this paper can extract curved symmetry axis and use global information. This gives the scheme some clear

advantages over previous ones, such as: 1) It can compute curved axis, 2) it provides connected axis, 3) it is remarkably stable to changes in the shape, 4) it provides a measure associated to the relevance of the axis in the shape, which can be used for shape description or for grouping based on symmetry or convexity 5) it can tolerate noisy and spurious data 6) it provides central points of the shape.

We have introduced the inertia surfaces and the tolerated length and we have shown how they can be used to find skeletons using a sophisticated version of the algorithm presented in [Sha'ashua and Ullman 88]. Similar measures might be used to find skeletons using other algorithms such as those presented in [Kass, Witkin and Terzopoulos 88] and [Zucker, Dobbins and Iverson 89].

We have also described how a part description of the shape can be obtained using the frame computation. We are currently incorporating this description scheme in a framework for early and middle level vision in which grouping and shape description are performed in a bottom up manner. The early vision modules provide a first guess as to what the most salient structures in the image are. Based on the grouping and shape description processes, certain salient structures are then selected in the image, and subsequent processing stages are applied selectively to the selected structures. This endows the system with a capacity that is similar to the use of selective attention in human vision. We are also investigating how the obtained description can be used for higher level vision tasks such as recognition.

The network presented in this paper computes skeletons in 2D images. The network can be extended to finding 3 dimensional skeletons from 3 dimensional data since the local estimates for orientation and curvature can be found in a similar way and the network extends to 3 dimensions, this, of course at the cost of increasing the number of processors. The problem of finding 3D skeletons from 2D images is more complex, however in most cases the projection of the 3D skeleton can be found by working on the 2D projection of the shape, especially for elongated objects.

The scheme presented in this paper can be extended to finding high, long and smooth curves in arbitrary surfaces. The scheme searches for the *best curve* using local estimates for orientation and curvature. The estimates can be obtained in an arbitrary surface by convolving it with oriented gabor filters at different orientations and scales. This could be applied to many tasks in vision. An example of such applications is finding dark blobs in images (see figure 9), the scheme selects both a region and a scale in the image.

Acknowledgements

Thanks to David Beymer, Thomas Breuel, Eric Grimson, Tomaso Poggio, Amnon Sha'ashua, Shimon Ullman and Woody Yang for useful discussions and suggestions for improving the presentation of the paper. Thanks to James Mahoney, Eric Saund and the staff of the Electronics Documents Laboratory at Xerox P.A.R.C. where I did the Connection Machine implementation of the scheme presented in this paper during summer 1989.

Appendix I

In the appendix we show that the set of possible saliency measures that can be computed with the network defined in [Sha'ashua and Ullman 88] (see also section 3) is limited.

Proposition 1 *The use of more than one state variable in the saliency network defined in section 3 does not increase the set of possible saliency functions that can be computed with the network.*

Proof: The notation used in the proof will be the one used in section 3. We will do the proof for the case of two state variables, the generalization of the proof to more state variables follows naturally. Each edge will have a saliency state variable $s_{i,j}$ and an auxiliary state variable $a_{i,j}$ and two functions to update the state variables: $s_{i,j}(n+1) = MAX_k \mathcal{F}(p, s_{j,k}(n), a_{j,k}(n))$ and $a_{i,j}(n+1) = \mathcal{G}(p, s_{j,k}(n), a_{j,k}(n))$. We will show that for any pair of functions \mathcal{F} and \mathcal{G} either they can be reduced to one function or there is a network for which they do not compute the optimal curves.

If \mathcal{F} does not depend on its last argument $a_{j,k}$ then the decision of what is the most salient curve is not affected by the introduction of more state variables so we can do without them. Observe that we might still use the state variables to compute additional properties of the most salient curve without affecting the actual shape of the computed curve.

If \mathcal{F} does depend on its last argument then there exists some p, x, y and $w \in \mathbb{R}$ such that: $\mathcal{F}(p, y, x) < \mathcal{F}(p, y, w)$. Assuming continuity this implies that there exists some $\epsilon > 0$ such that: $\mathcal{F}(p, y - \epsilon, x) < \mathcal{F}(p, y, w)$. Assume now two curves of length n starting from the same edge $e_{i,j}$ such that $s_{1,j}(n) = y$, $a_{1,j}(n) = x$, $s_{2,j}(n) = y - \epsilon$ and $a_{2,j}(n) = y$. If the algorithm were correct at iteration n it would have computed the values $s_{1,j}(n) = y$, $a_{1,j}(n) = x$ for the variables $s_{i,j}$ and $a_{i,j}$. But then at iteration $n+1$ the saliency value computed for an edge $e_{h,i}$ would be $s_{h,i} = \mathcal{F}(p, y - \epsilon, x)$ instead of $\mathcal{F}(p, y, w)$ that corresponds to a curve with a higher saliency value. \square

References

- [1] C. Arcelli, L.P. Cordella, and S. Levialdi. From local maxima to connected skeletons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(2):134-143, 1981.
- [2] S.C. Bagley. Using models and axes of symmetry to describe two-dimensional polygonal shapes. Master's thesis, Massachusetts Institute of Technology, 1985.
- [3] H. Blum. A transformation for extracting new descriptors of shape. In Walthen Dunn, editor, *Models for the perception of speech and visual form*, pages 362-380. MIT Press, Cambridge, MA, 1967.
- [4] H. Blum and R.N. Nagel. Shape description using weighted symmetric axis features. *Pattern Recognition*, 10:167-180, 1978.

- [5] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34:344-371, 1986.
- [6] M. Brady and H. Asada. Smoothed local symmetries and their implementation. *International Journal of Robotics Research*, 3(3):36-61, 1984.
- [7] M. Brady and G. Scott. Parallel algorithms for shape representation. In Ian Page, editor, *Parallel Architectures and Computer Vision*. OUP, 1988.
- [8] J.H. Connell. Learning shape descriptions: generating and generalizing models of visual objects. Technical Report AI TR No. 853, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1985.
- [9] J.H. Connell and M. Brady. Generating and generalizing models of visual objects. *Artificial Intelligence*, 31:159-183, 1987.
- [10] R. Duda and P. Hart. *Pattern classification and scene analysis*. Willey, 1973.
- [11] M.M. Fleck. Local rotational symmetries. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 332-337, 1986.
- [12] M.M. Fleck. Boundaries and topological algorithms. Technical Report AI-TR-1065, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1988.
- [13] G. Kanizsa and W. Gerbino. Convexity and symmetry in figure-ground organization. In M. Hele, editor, *Vision and Artifact*. Springer, New York, 1976.
- [14] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, pages 321-331, 1988.
- [15] C. Koch and S. Ullman. Shifts in selective visual attention: Towards the underlying neural circuitry. *Human Neurobiology*, 4:219-227, 1985.
- [16] J.V. Mahoney. Image chunking: defining spatial building blocks for scene analysis. A.I. Technical Report No. 980, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1987.
- [17] R. Nevatia and T.O. Binford. Description and recognition of curved objects. *Artificial Intelligence*, 8:77-98, 1977.
- [18] A. Rosenfeld and J.L. Pfaltz. Distance functions on digital pictures. *Pattern Recognition*, 1:33-61, 1968.
- [19] G.L. Scott, S.C. Turner, and A. Zisserman. Using a mixed wave-diffusion process to elicit the symmetry set. *Image and Vision Computing*, 7(1):63-70, 1989. This paper appeared also in the Proceedings of the 4th Alvey Vision Conference, Manchester, England, 1988.
- [20] J. Serra. *Image Analysis And Mathematical Morphology*. Academic Press Inc., London, 1982.
- [21] A. Sha'ashua and S. Ullman. Structural saliency: The detection of globally salient structures using a locally connected network. In *Proceedings of the International Conference on Computer Vision*, pages 321-327, 1988.
- [22] S.W. Zucker, A. Dobbins, and L. Iverson. Two stages of curve detection suggest two styles of visual computation. *Neural Computation*, 1(1):68-81, 1989.

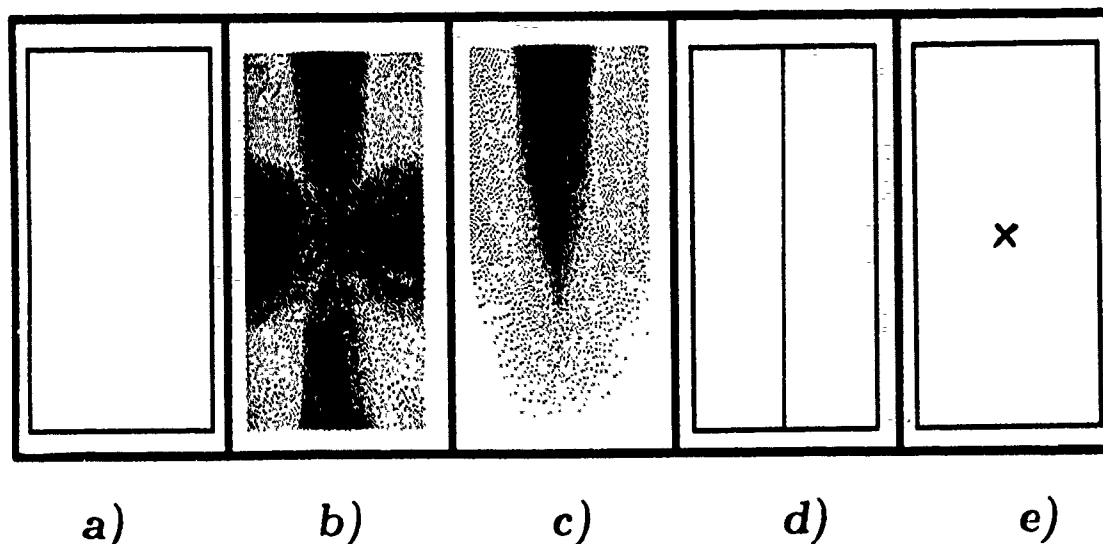


Figure 10: a) Rectangle. b) Skeleton sketch for the rectangle. Circles along the contour indicate local maxima in the skeleton sketch. c) Skeleton sketch for the rectangle for one particular orientation, vertical-down in this case. d) Most salient curve. e) Most interesting point for the most salient curve.

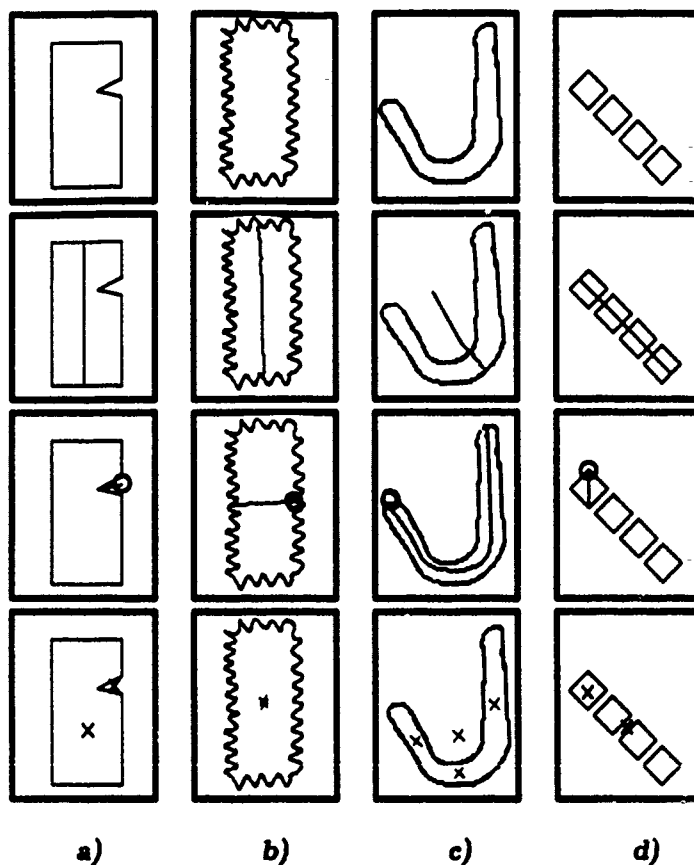


Figure 11: Top. Four shapes: a notched square, a stamp, a J, and Mach's demonstration. Second row: The most salient curve found by the network for each of them. Observe that the scheme is very stable under noisy or bent shapes. Third row: The most salient curve starting inside the shown circles. For the J shape the curve shown is the most salient curve that is inside the shape. Fourth row: The most interesting point according to the curves shown in the two previous rows. See text for details.

Indexing Via Color Histograms

Michael J. Swain and Dana H. Ballard

Computer Science Department

University of Rochester

Rochester, NY 14627, USA

swain@cs.rochester.edu, ballard@cs.rochester.edu

Abstract

The color spectrum of multicolored objects provides a robust, efficient cue for indexing into a large database of models. This paper shows color histograms to be stable object representations over change in view, and demonstrates they can differentiate among a large number of objects. It introduces a technique called *Histogram Intersection* for efficiently matching model and image histograms. Color can also be used to search for the location of an object. An algorithm called *Histogram Backprojection* performs this task efficiently in crowded scenes.

1 Introduction

Computer vision is moving into a new era in which the aim is to develop visual skills for robots that allow them to interact with a dynamic, unconstrained environment. To achieve this aim, new kinds of vision algorithms need to be developed which run in real time and subserve the robot's goals. Two fundamental goals are determining the identity of an object with a known location, and determining the location of a known object. Color can be successfully used for both tasks.

Color has been neglected recently as a recognition cue, although it has been used in earlier work [Ohlander *et al.*, 1978]. One reason for this may have been the lack of good algorithms for *color constancy*, that is, perceiving a stable perception of color over varying light conditions, as people do in most circumstances. However, recently there has been great progress in correcting for both the chromaticity of the illuminant [Maloney and Wandell, 1986, Rubner and Schulten, 1989] and for geometric effects such as specularity [Klinker *et al.*, 1988]. Given that reasonable color constancy can be achieved, color has enormous value in recognition because it is a local surface property that is view invariant and largely independent of resolution. Shape cues, by contrast, are highly resolution dependent, and only a highly restricted set are view invariant (e.g. corners, zeros of curvature).

Perhaps another reason that color has not been used is that it is not intrinsically related to the object's identity

in the way that other cues, e.g., form, are. This view is well represented by Biederman [1985]:

"Surface characteristics such as color and texture will typically have only secondary roles in primal access ... we may know that a chair has a particular color and texture simultaneously with its volumetric description, but it is only the volumetric description that provides efficient access to the representation of CHAIR."

However, this opinion is easily challenged. There are many examples from nature where color is used by animals and plants to send clear messages of enticement or warning. The manufacturing sector uses color extensively in packaging to market goods. Robotic vision systems can also use representations that are heavily personalized to achieve efficient behaviors. For example, it may not be helpful to model coffee cups as being red and white, but *yours* may be, and that color combination is very useful in locating it.

1.1 What vs. Where

A significant feature of the gross organization of the primate visual brain is the specialization of the temporal and parietal lobes of visual cortex. The parietal cortex seems to be subserving the management of locations in space whereas the temporal cortex seems to be subserving the identification of objects in the case where location is not the issue. In a striking experiment by Mishkin [1987], monkeys with parietal lesions fail at a task that requires using a relational cue but have no trouble performing a very similar task that requires using a pattern cue. The reverse is true for temporal lesions. Why should the primate brain be specialized in this way? If we think generally about the problem of relating internal models to objects in the world, then one way to interpret this "What/Where" dichotomy is as a suggestion that image interpretation, the general problem of associating many models to many parts of the image simultaneously, is either too hard or unnecessary, or both (see Table 1.1). In order to build vision systems which function in real-time, perhaps the problem must be simplified.

The approach taken in Section 2 is to answer the question "What" assuming that the approximate location of the object is known. This is done by using a color histogram as the representation, which counts how much of each color occurs in the image.

		Object to Match Against	
		One	Many
Image Portions	One		Identification: trying to identify an object whose location can be fixated
	Many	Location: trying to locate an object whose identity is known	Image interpretation: Too hard?

Table 1: The biological organization of cortex into What/Where modules may have a basis in computational complexity. Trying to match a large number of image segments to a large number of models at once may be too difficult.

Section 3 shows how a model histogram can be back-projected onto an image to solve one aspect of the "Where" problem, which is the location of an object of known identity in the image. Again, the view invariance of color precludes the calculation of orientation but simplifies the algorithm enormously.

1.2 Color Histograms

Given a discrete color space defined by some color axes (e.g. red, green, blue), the color histogram is obtained by counting the number of times each color occurs in the image array. To illustrate, Figure 1 (page 4) shows the output from a color camera together with a color histogram obtained from the image.

Histograms are invariant to translation and rotation about an axis perpendicular to the image plane, and change only slowly under change of angle of view, change in scale and occlusion. Because histograms change slowly with view, a three-dimensional object can be adequately represented by a small number of histograms, corresponding to a set of canonical views [Koenderink and van Doorn, 1976].

Histograms are efficient to compute using image processing hardware. Generating a histogram from a 512 x 485 image takes about 40 milliseconds using a MaxVideo FeatureMax board, including the time needed to transfer the histogram to the host.

Both the object identification and object location implementations described in the following sections use color histograms to represent objects.

2 Histogram Intersection

Because the model database may be large, we can only afford a highly restricted amount of processing per model, but at the same time we must be able to overcome the problems that hinder recognition, most importantly

- distractions in the background of the object,
- viewing the object from a variety of viewpoints,
- occlusion,
- varying lighting conditions.

The matching method proposed here, called *Histogram Intersection*, is robust to the first three problems; the last is left to a color constancy module that operates on the input prior to the histogram stage. Histogram Intersection is also extremely efficient and easy to implement.

Section 2.1 describes the algorithm. Section 2.2 shows that histogram intersection is capable of differentiating objects from a large database.

2.1 Description

Given a pair of histograms, I (image) and M (model), each containing n buckets, the intersection of the histograms is defined to be

$$\sum_{j=1}^n \min(I_j, M_j).$$

The result of the intersection of a model histogram with an image histogram is the number of pixels from the model that have corresponding pixels of the same color in the image. To obtain a fractional match value between 0 and 1 the intersection is normalized by the number of pixels in the model histogram. The match value is then

$$\frac{\sum_{j=1}^n \min(I_j, M_j)}{\sum_{j=1}^n M_j}.$$

The Histogram Intersection match value is not reduced by distracting pixels in the background. This is the desired behavior since complete segmentation of the object from the background is difficult to guarantee. The histogram intersection match value is only increased by a pixel in the background if

- the pixel has the same color as one of the colors in the model, and
- the number of pixels of that color in the object is less than the number of pixels of that color in the model.

Histogram Intersection is robust to scale changes but not scale invariant. However, there are a number of ways of determining the approximate depth of an object, from laser or sonar range finders, disparity, focus or touching the object with a sensor. The depth value combined with the known size of the object can be used to scale the model histogram. Alternatively, if it is possible to segment the object from the background and it is not significantly occluded the image histogram can be scaled to be the same size as the model histogram.

The Histogram Intersection approach to histogram matching can be related to classical pattern recognition by considering each bin in the color histogram as a feature. An object is then a point in an n -dimensional space, where n is the number of bins in the histogram. If histograms are scaled to be the same size, then Histogram Intersection is equivalent to a scaled sum of absolute differences and therefore defines a distance metric. If histograms are not the same size, as when the scaling is done by distance, then Histogram Intersection is not

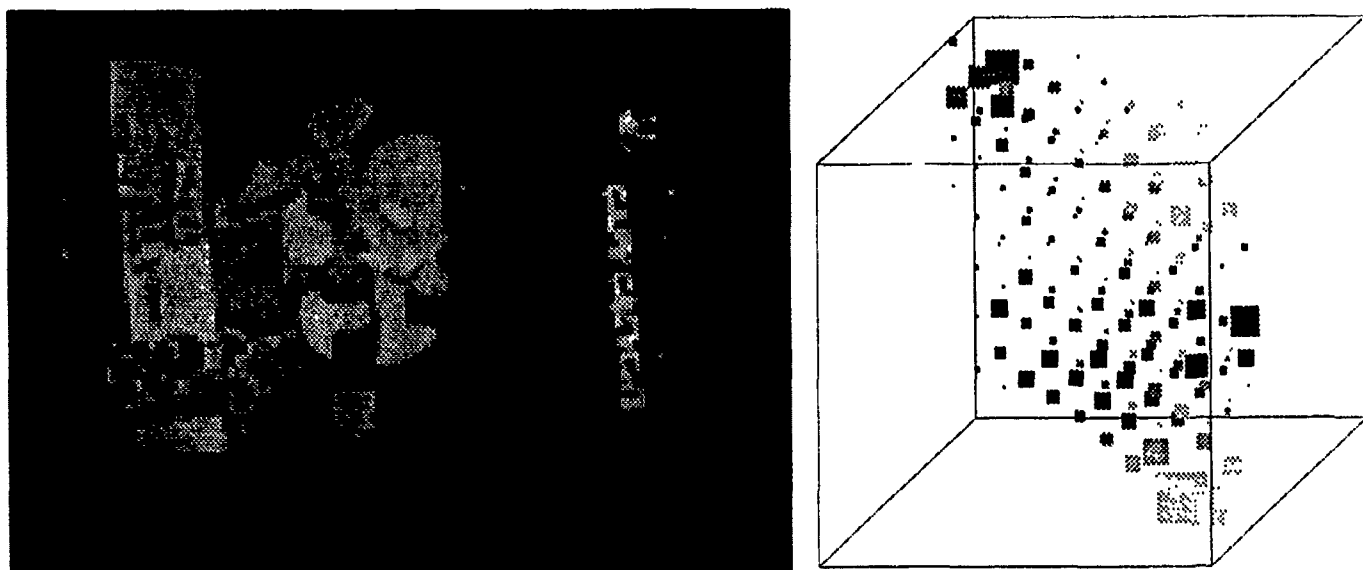


Figure 1: Left: Image of a Crunchberries cereal box. Right: Three dimensional color histogram of the Crunchberries image with the black background subtracted.

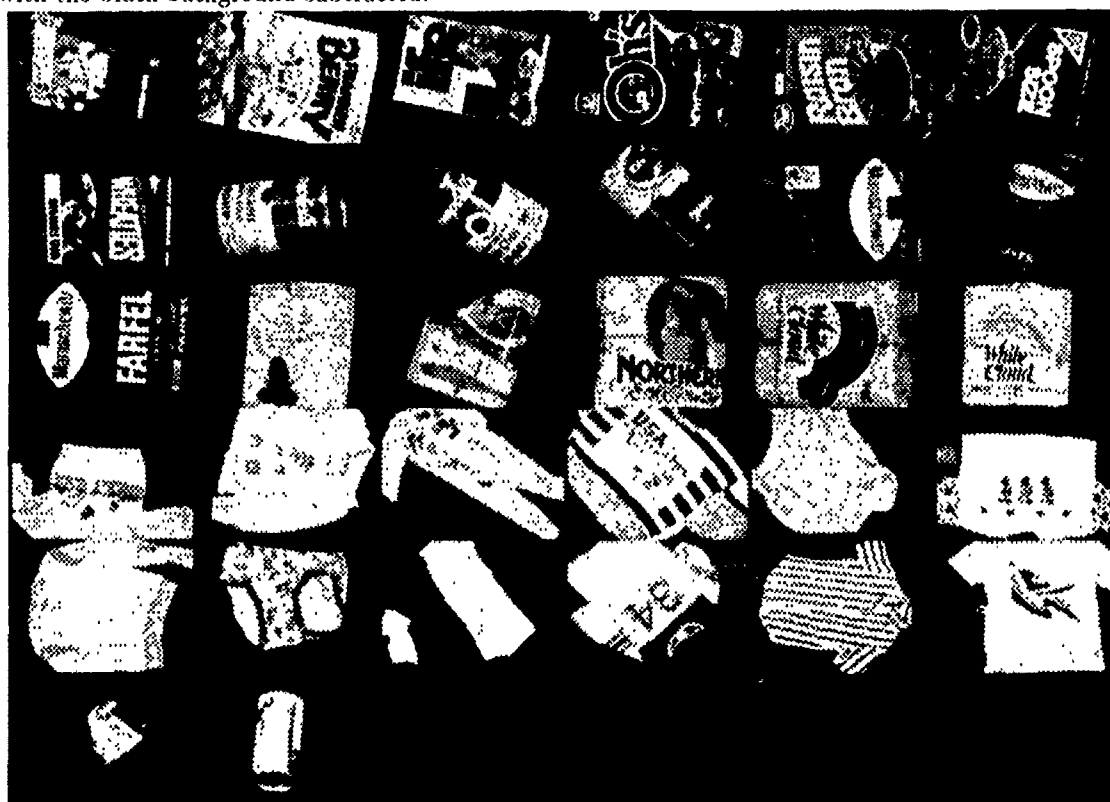


Figure 2: Model indexing experiment based on color cues (continued in the next two figures). Each of the images of unknown objects shown here is identified with the model color histogram (next figure) that best matches its own color histogram. Compared to the models the unknown objects are translated (Ajax), rotated about various axes (Frankenberry, Ajax), scaled (USA Flyer), occluded (Charmin), partly outside of the field of view (red, white striped shirt), and deformed (Mickey Mouse underwear).



Figure 3. The model database. Each of the sixty-six models shown here is represented by its color histogram

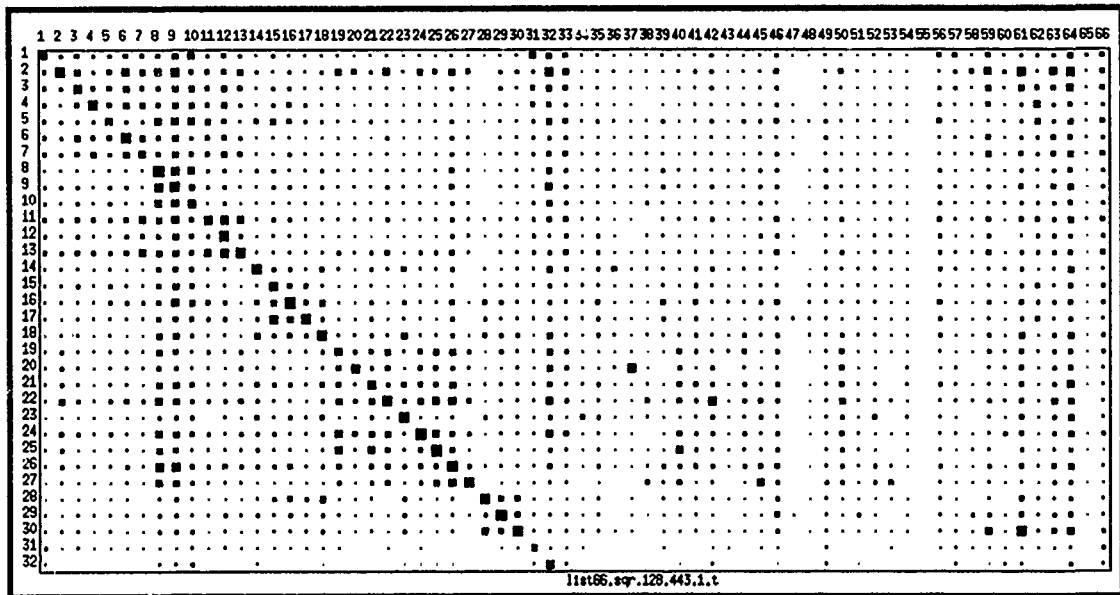


Figure 4. The results of matching all combinations of image and database histograms displayed pictorially where the size of the squares are proportional to match values. The dominance of the diagonal values shows that the correct match is almost always selected. Twenty nine of thirty-two matches are correct; in three cases the correct model received second highest score. Models are along the horizontal axis; unknown objects along the vertical axis.

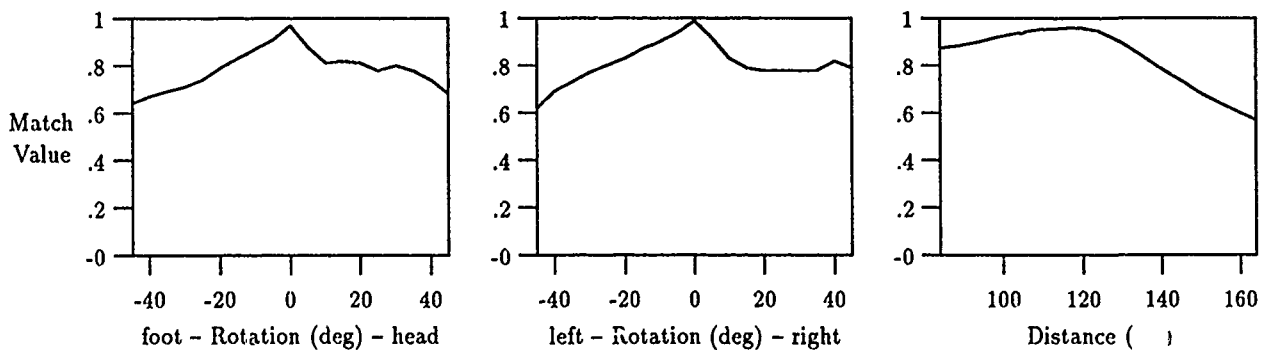


Figure 5. Variation of the Histogram Intersection match value as the camera is moved with respect to a S In the Distance graph the model image was taken at a distance of 124 cm.

Recognition Times (milliseconds)			
	Database Size		
	19	37	70
Histogram Intersection	38	73	150
Incremental Intersection	15	15	15

Table 2: Recognition times as a function of database size for the standard algorithm Histogram Intersection and the fast indexing scheme Incremental Intersection, using the 10 largest image bins. Recognition accuracy was equivalent for both algorithms. Timings were made on a SUN SPARCstation 1.

a metric because of the asymmetry between images and models.

Histogram Intersection is an efficient way of matching histograms. Its complexity is linear in the number of elements in the histograms. Two 16x16x8 histograms can be matched in 2 milliseconds on a SUN Sparcstation 1 (a 12 MIP RISC machine).

2.2 Experimental Results

An experimental test of histogram intersection suggests that the technique is capable of differentiating among a large database. For the 66 object database shown in Figures 3-4, the correct model is the best match 90% of the time and is always one of the top two matches. Other, more expensive, matching techniques can be used to verify which of the top scoring models is the correct one, so it is not crucial that the correct model is always the best match. In the experiment the models were segmented from the background prior to generating the model histograms. No segmentation was performed on the images of the unknown objects.

Figure 5 shows how the Histogram Intersection match value changes as the camera is rotated about the Snoopy Doll shown in Figure 3, and moved closer and further from the doll. Compare these match values to the ones in Figure 6. Even at 45 degrees rotation or 1 1/3 times the original distance the match value (about 0.6) is higher than 99 percent of the false matches. Thus, a small number of histograms may be used to represent a three-dimensional object.

The results of experiments which show that histogram intersection is insensitive to occlusion, and image and histogram resolution have been reported in [Swain, 1990]

Most of the information needed for identification is carried by large buckets in the histograms. An algorithm called Incremental Intersection takes advantage of this fact to index extremely efficiently into very large databases without sacrificing accuracy [Swain, 1990]. Although still linear in the size of the database, the constant of proportionality is extremely low. The efficiency of Histogram Intersection and Incremental Intersection is compared in Table 2.2.

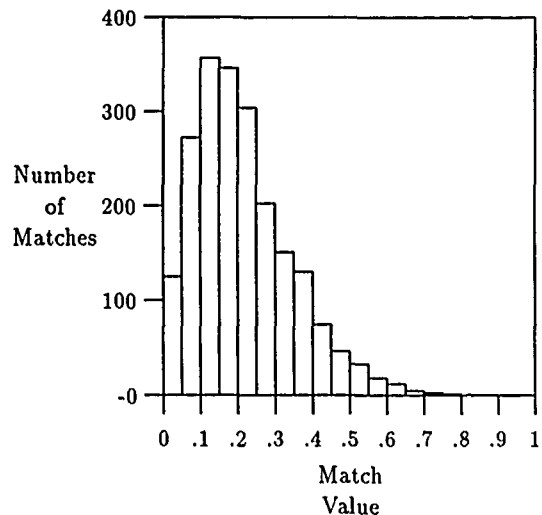


Figure 6: Distribution of match values for incorrect image - model matches for the models and images shown in Figures 3 and 2. The values counted here are all the off-diagonal elements of the matrix shown in Figure 4.

3 Object Location via Histogram Backprojection

The previous sections discussed recognizing an unknown object whose location is known, the "Identification" box in Table 1.1. This section discusses the complementary task, locating a known object, the "Location" box in the same table. This task can also be accomplished using color histograms and an algorithm called *Histogram Backprojection*.

Histogram Backprojection answers the question "Where are the colors in the image that belong to the object being looked for (the *target*)?" The answer is given in such a way so that the colors that appear in other objects besides the target are deemphasized so that they are less likely to distract the search mechanism. Experiments show that the technique works for objects in cluttered scenes under realistic conditions.

As in Histogram Intersection, in Histogram Backprojection the model (target) is represented by its multidimensional color histogram M . The histogram of the image, I , is also computed and a third histogram R , which is the ratio of M divided by I , is computed. It is this histogram R which is backprojected onto the image, that is, the image values are replaced by the values of R that they index. The backprojected image is then convolved by a mask, which for compact objects of unknown orientation could be a circle with the same area as the expected area subtended by the object. The peak in the convolved image is the expected location of the target, provided the target appears in the image.

More precisely, let $h(c)$ be the histogram function which maps a color c (a three-dimensional value) to a histogram index (another three-dimensional value). Let D^r be a disk of radius r :

$$D^r_{x,y} = \begin{cases} 1 & \text{if } \sqrt{x^2 + y^2} < r \\ 0 & \text{otherwise} \end{cases}$$



Figure 7. Results of step 1 of Histogram Backprojection, using Figure 2 as the image and the striped blue and white shirt as the target. The blue hue is found only a small area outside of the target, so it gives a strong response. White is found in many objects so it gives a weak response.

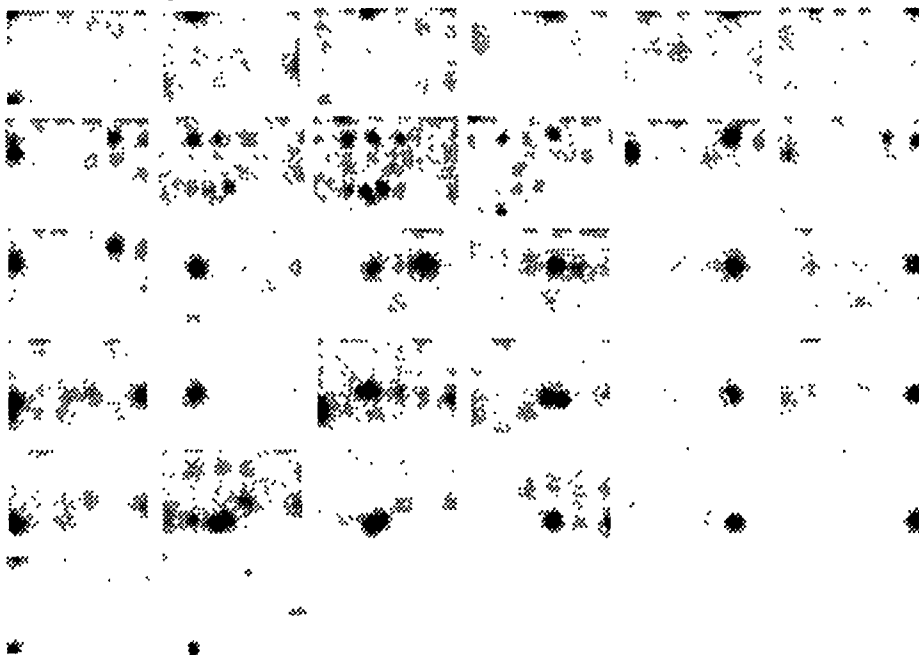


Figure 8. Results of step 2 of Histogram Backprojection, for the same image as above. The results for all the models are shown in the image, each in the rectangle corresponding to the location of that model in the composite photo. When the algorithm successfully finds the object, the darkest black dot in the small images is in the same location within that image as the image is in the composite.

Define the *index* function to return a pixel (x, y) with the value of its argument, and let the $*$ symbol denote convolution. Then Histogram Backprojection can be written:

1. for each i, j, k do

$$R(i, j, k) := \frac{M(i, j, k)}{I(i, j, k)}$$
2. for each x, y do

$$b_{x, y} := \min (R(h(c_{x, y})), 1)$$
3. $b := D^r * b$
4. $(x_t, y_t) := \text{index}(\max_{x, y} b_{x, y})$

As a demonstration of Histogram Backprojection, we consider Figure 2 as a single crowded scene, and look for objects within it using the models from Figure 3. The results are shown in Figures 7 and 8. In all cases but four the largest peak in the convolved image corresponds to the correct object. The four cases in which it doesn't are listed below in the format (target: objects receiving larger response).

1. Wheaties: Matzo Farfel.
2. Campbell's Clam Chowder: red and white shirt, Campbell's chicken soup.
3. Charmin: orange White Cloud.
4. Mickey Mouse underwear: red and white shirt, USA Flyer.

Because the convolution can be carried out on a reduced resolution image, *Histogram Backprojection* is very efficient. It has been implemented on a MaxVideo image processor, on which it runs in real time (0.1 seconds per frame).

4 Previous Work

Early work on classification using color invariably assumed the objects were monochromatic [Ohlander *et al.*, 1978]. Recently [Wixson and Ballard, 1989] used color to find multi-colored objects with a mobile camera. Their match technique is histogram-based, like Histogram Intersection, and influenced the development of the match techniques in this paper. Because they desired a scale invariant match value it has important differences. They compare ratios of peaks in the model and image histograms. This technique fails for uni-colored objects, and it can suffer severely under occlusion which cuts across most of the colored regions, or by the presence of another object in the image whose histogram peaks overshadow some of the peaks of object being matched.

5 Summary

The main points of this paper are:

1. Color histograms are a robust, efficient representation for indexing into a database of a large number of everyday colored objects.
2. Image and model histograms can be compared by a method called *Histogram Intersection*, which asks how many of the pixels in the model histogram are found in the image.

3. An incremental version of Histogram Intersection, called *Incremental Intersection*, allows fast indexing into a large database without sacrificing accuracy.
4. Multi-colored objects can be located in a crowded scene using a technique called *Histogram Backprojection*.

Because of color's important applications and ease of use, color cameras and digitizing facilities should be a feature of robotic systems that have to operate in normal human surroundings.

6 Acknowledgments

Lambert Wixson, Leo Hartman, Randal Nelson and the Rochester vision group were a source of influential discussions and ideas.

References

- [Biederman, 1985] I. Biederman. Human image understanding: Recent research and a theory. *Computer Vision, Graphics and Image Processing*, 32(1):29-73, 1985.
- [Klinker *et al.*, 1988] Gudrun J. Klinker, Steven A. Shafer, and Takeo Kanade. The measurement of highlights in color images. *International Journal of Computer Vision*, 2:7-32, 1988.
- [Koenderink and van Doorn, 1976] J. J. Koenderink and A. J. van Doorn. The singularities of the visual mapping. *Biological Cybernetics*, 24:51-59, 1976.
- [Maloney and Wandell, 1986] Laurence T. Maloney and Brian A. Wandell. Color constancy: A method for recovering surface spectral reflectance. *Journal of the Optical Society of America A (JOSA-A)*, 3(1):29-33, January 1986.
- [Mishkin and Appenzeller, 1987] Mortimer Mishkin and Tim Appenzeller. The anatomy of memory. *Scientific American*, 1987.
- [Ohlander *et al.*, 1978] R. Ohlander, K. Price, and D.R. Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, 8:313-333, December 1978.
- [Rubner and Schulten, 1989] J. Rubner and K. Schulten. A regularized approach to color constancy. *Biological Cybernetics*, 61:29-36, 1989.
- [Swain, 1990] Michael J. Swain. *Color Indexing*. PhD thesis, University of Rochester, 1990.
- [Wixson and Ballard, 1989] Lambert E. Wixson and Dana H. Ballard. Real-time detection of multi-colored objects. In *SPIE Sensor Fusion II: Human and Machine Strategies*, volume 1198, November 1989.

Real-time Qualitative Detection of Multi-colored Objects for Object Search

Lambert E. Wixson*

Computer Science Department

University of Rochester

Rochester, NY 14627

email: wixson@cs.rochester.edu

Abstract

This paper considers the task of using a mobile camera to search for a specified object in a room. We call this the *object search* task. Introspection reveals that humans perform such tasks countless times every day. However, despite its pervasive nature, object search has been the target of little research effort. The paper begins with a general discussion of the object search problem. Given that the goal of object search is to reliably find the desired object with a minimum of effort, we identify three main abilities that any object search system must possess. These are the abilities to apply multi-stage object recognition strategies, to reason about occlusion, and to use high-level knowledge about spatial contexts to predict likely locations of objects.

A preliminary implementation of object search is presented that illustrates a method for real-time detection of the presence of known *multi-colored* objects in a scene. The method is based on the assumption that the color histogram of an image can contain object "signatures" which are invariant over a wide range of scenes and object poses. The resulting algorithm has been easily implemented and used to build a robot that can direct its gaze over a room searching for an object.

1 Introduction

Vision and robotics research has investigated many high-level visual tasks, such as object recognition, exploration to acquire a world model, navigation, and obstacle avoidance. These problem areas were identified long ago because of their usefulness for autonomous functioning, humans use these abilities almost constantly. This paper describes a visual task, the *object search* task [Wixson, 1990b], that is also constantly performed by humans but has been previously unexplored. The task is simply stated: "Using a mobile camera that can move in some delimited 3-D space, such as a room, find a specified object¹ that is somewhere in the space."

The first half of this paper contains a general discussion of the object search problem. Motivations for the study of object search are presented, related work is discussed, and the major issues involved in any searching task are highlighted. A preliminary implementation of object search is presented that illustrates a method for quickly detecting the presence of known *multi-colored* objects in a scene. The method is based on the assumption that the color histogram of an image can contain object "signatures" which are invariant over a wide range of scenes and object poses. The resulting algorithm has been easily implemented and used to build a robot that can sweep its gaze over a room searching for an object.

2 Why study object search?

To motivate the study of object search, let us first consider the computational benefits that a robot might gain from a fast and robust object search system. The most important of these is that possession of fast object search capabilities obviates the need for the robot to construct a detailed world model. There is no need for the robot's perceptual system to attempt to identify and store (for future reference) the exact locations of every object in every image it sees while functioning. Instead, when the robot needs to find an object, it can simply use its search system. Not only does this allow the robot to avoid the computationally intractable task of image interpretation in order to identify objects in each image [Swain *et al.*, 1989], but it also eliminates the need to determine a

*This research was supported by DARPA US Army Engineering Topographic Laboratories Contract DACA76-85-C-001, and NSF Institutional Infrastructure grant CDA-8822724.

¹This paper will refer to the object being searched for as the *target object* or *desired object*.

world coordinate system in which to represent the objects. Such coordinate systems are often impractical due to sensor and effector error [Brooks, 1987]. As a result of examining fixation traces of subjects instructed to remember the position of objects in a room, Ballard [1989] has conjectured that humans do not construct a detailed world model routinely. It is tempting to speculate that humans compensate by using their object search abilities.

Besides allowing a robot to simplify its world representation, object search allows some simplifications in its own implementation. These simplifications arise from the assumption that some higher cognitive process determines a *single* desired object, this knowledge of the desired object allows the choice of appropriate recognition algorithms and the selection of camera viewing parameters that facilitate the chosen algorithm. No object recognition algorithm is applicable to all types of objects. The search system may have to choose between using a shape-based recognition technique, a recognition strategy based on surface properties such as color or texture, or a method based on "recognition by distinguishing features" [Garvey, 1976]. Knowing the object to be recognized allows a proper choice to be made.

Once a recognition method has been selected, a robust perceptual system must be able to select viewing parameters that allow the method to function properly. Doing so involves not only knowledge about the abilities of the recognition module, but also knowledge about the object that it is trying to recognize. For example, the field-of-view must be large enough to allow a significant part of the desired object (enough to allow its recognition) to fit in the field, yet small enough that the detail needed by the recognition module is readily visible. Since an object search system knows the size of the object it is to identify, it can choose appropriate viewing parameters as a function of the depth of the scene being imaged.

When one recognizes the above-described benefits of object search and the implementational simplifications stemming from its goal, the utility of an object search system seems very significant.

3 Related work

Although there have been several projects whose goal was to explore an area in order to construct a depth map of the area [Krotkov, 1987] and to identify the objects in the area [Bolle *et al.*, 1989], the problem of building robots that search a room for a specified object appears to have received almost no attention. The only work in this area is the MIT AI Lab's robot that roams an area searching for and collecting soda cans [Connell, 1989]. This research, however, has concentrated more on the issues involved in constructing robots with a subsumption architecture than on object search issues. Its treatment of issues such as reasoning about occlusion and using high-level knowledge to direct the sensors has been very limited.

Feldman and Sproull [1977] advocated a decision-theoretic approach to planning, and suggested the use of decision theory to guide a robot searching for objects needed in a plan, but did not consider object search fur-

ther. In addition, search theory, a subfield of operations research, has produced a large body of knowledge concerning optimal search strategies for a wide variety of situations [Ahlswede and Wegener, 1987].

Unlike the general problem of object search, the task of searching for a specific object in a *single image* has received a great deal of attention. This is simply the *object recognition* task. Work in this area will be reviewed in Section 4.1.

Like robotics researchers, cognitive psychologists appear to have neglected object search. However, they have devoted considerable effort to the problem of finding a specific object or stimulus in a single image, known as the *visual search* problem [Biederman *et al.*, 1973, Treisman, 1988, Rabbitt, 1978, Visual Search, 1973, Morris and Horne, 1959]. In addition, much work [Yarbus, 1967, Fisher *et al.*, 1981] has studied eye movements, known as *foveation sequences* or *fixation sequences*. Various researchers have studied separately eye movements in visual search in images without context [Cohen, 1981], the effects of context on eye movements in exploration tasks [Antes and Penland, 1981], and modeling of eye movement strategies [Stark and Ellis, 1981, Rimey and Brown, 1990]. Experiments now need to be performed which build upon these studies by studying and modeling eye movements used in visual search in context-rich pictures.

4 Research issues in object search

The previous sections have discussed the lack of work on object search and have attempted to motivate study in this area. Let us now consider the major issues in building an object search system. The search system should be robust (*i.e.* should usually find the object) and, *just as importantly*, be fast. The importance of robustness is obvious. The importance of speed stems from Section 2's hypothesis that a fast object search system can be used to eliminate the problem of identifying every object and storing its coordinates for future reference. The speed of object search is affected by two factors - the number of gazes considered (*i.e.* the number of images examined) and the speed with which a single image is processed. This section explores the three main issues that arise from our concern with robustness and efficiency, concluding with a discussion of issues in formulating these issues within a single decision-theoretic framework.

4.1 Object Recognition

A major component of any object search system is the set of methods used to recognize the desired object. The task of searching for a specific object in a single image is simply the *object recognition* task. We use intensity image data (as opposed to 3-D range data) for recognition due to the fact that there are many different objects in the world, such as books, magazines, and packaged goods that have the same 3-D shape but are distinguished by different surface markings.

Object recognition methods fall into two categories. The first is the set of *model-based* strategies that attempt to match the 2-D image data to an internal 3-D model [Goad, 1987, Lowe, 1985, Lamdan and Wolfson, 1988,

Huttenlocher, 1988, McIvor, 1988, Lowe, 1989]. Although these methods are effective, they require an internal model of the object and are often restricted to rigid objects. More seriously, these methods calculate the pose of the object as part of the recognition process, requiring times polynomial in the number of image and model features. As a result, in practice they are slowed considerably. The typical time to process a single image (not counting edge extraction, which can be performed in real time) is between 25 seconds and 5 minutes [McIvor, 1988, Huttenlocher, 1988, Huttenlocher, 1989]. These times are significant when one considers the possibility that many images might have to be considered in the course of a single search.

The second class of object recognition methods is known as *non-correspondence matching*. These methods do not attempt to compute the pose of the object in the matching process, instead relying on surface properties such as color or texture [Garvey, 1976, Swain, 1990]. Surface properties are particularly useful when attempting to match objects that are hard to model. Non-correspondence methods based on surface properties also have the advantage of being quickly computable with local methods. Unfortunately, they are not always robust with respect to occlusion, rotation of the object, or changes in illumination.

Non-correspondence and model-based methods can be used in complementary ways in object search. For example, surface color is not destroyed by a wide field-of-view that reduces the scale of objects in the image, whereas the edges that can be extracted from the image and input to a model-based algorithm might be corrupted or discarded as noise due to such scaling. In the case where one is searching for a colored object, this suggests a two-stage approach. In the first stage, a wide field-of-view might be used, and in each image "blobs" whose color is the same as that of the desired object might be extracted. The camera could then zoom in on each such blob in order to produce an image suitable for processing with a model-based method. Sections 5, 6, and 7 describe an implementation of a similar strategy that in the first stage uses an object *detection* algorithm to estimate the likelihood of the presence of the target object in each image.

4.2 Reasoning about occlusion

In order to robustly find objects, the system must have not only a robust object recognition system, but also a mechanism for reasoning about occlusion. If the desired object has not been found by a set of camera orientations about a fixed location, the camera must be moved to a different point in order to view as-yet-unexamined regions of space. This means that the system must be able to identify those volumes in the scene which have not yet been viewed and must be able to choose a new viewing position from which one or more of these volumes can be examined.

The problem thus stated implies that the object search system, if it ran forever, would eventually examine all volumes of the entire scene. This is similar to the goals of robot exploration and mapping [Bolle *et al.*, 1989,

Krotkov, 1987]. However, unlike the majority of this work, there are no goals of constructing a qualitative model of the connectivity of the environment or of building a map of the objects that comprise the scene. The representation constructed needs only to support reasoning about unexamined areas. It is our hope that this simplified goal will allow a simpler representation to suffice.

4.3 Indirect Search

The previous discussion of the cost of model-based object recognition illustrated the importance of minimizing the number of images considered in the course of the search. One obvious step towards achieving this goal is to use high-level knowledge to predict likely locations of objects and to order the camera gazes considered so as to minimize the time to find the object. Garvey [1976] referred to this use of high-level knowledge as *indirect search*.

There are several ways in which high-level knowledge can guide search. One method, if the poses of certain objects are known, is to use these locations to predict likely locations for the desired object. For example, suppose the location of a desk is known in advance. If the robot is then told to search for a chair, it could start by searching near the desk. Moreover, if the robot knows there is also a typewriter located next to the desk, it should start its search by examining the space which is near both the typewriter and the desk. We call this the *location constraint* method.

A second method, called *detectability-driven search*, that requires no initial knowledge of poses, determines whether there are objects that are more easily detectable than the desired object (perhaps due to color, size, or distinguishing geometric features) and if they can be found, are likely to provide information about the relative location of the target object. If so, then these objects are searched for, and once these are found then location constraint may be applied. For example, when searching for a small object such as a pencil, it may be advantageous to first search for a larger object which is likely to constrain the location of the pencil, such as a desk.

Probabilistic representations of high-level knowledge that facilitate these search methods as well as the learning of this knowledge have been discussed in [Wixson, 1990a].

4.4 Control Strategies - Putting it all together

The previous discussion has highlighted three important aspects of object search. Let us now consider how these topics can be integrated to form a complete search system. Decision theory provides an elegant framework in which to handle our dual goals of robustness and efficiency, and previous researchers have used decision theory for topics related to object search and active vision [Garvey, 1976, Feldman and Sproull, 1977, Bajcsy, 1988]. Efficiency issues as well as the speed of the search and various costs of execution can be combined into a single cost function, and measures of detectability of the target object such as its size or color can be grouped into a function that predicts the probability of actually finding the object. This allows the formalization

of the problem as an attempt to choose a search strategy with minimal expected cost. Although not discussed in this paper, we are currently attempting to formulate an entire object search system decision-theoretically.

5 Preliminary Implementation

To carry out object search experiments, we have mounted a camera platform on a Puma robot arm [Brown, 1988]. The arm is mounted in the center of a 16' x 24' room that contains cluttered scenes containing everyday objects. Our preliminary implementation has neglected the issues of selecting appropriate fields-of-view, reasoning about occlusion, and indirect search, concentrating initially on studying the tradeoffs between non-correspondence and shape-based object recognition techniques. A simple search strategy is used - the camera is positioned in the center of the room and rotated 360 degrees in increments of 15 degrees, with the field-of-view of the camera adjusted so that the spatial volumes seen by adjacent increments are spatially adjacent. This 360 degree rotation is executed for each of several pitch angles, so that the camera can examine the upper walls, the lower walls, and the floor.

Unfortunately, this results in a total of 72 images that must be examined individually to determine whether they contain the target object. With run times of at least 25 seconds per image, shape-based recognition methods cannot be invoked for each of these images. A mechanism is needed that can prune out many of the possible images upon which a shape-based algorithm would be invoked, leaving only a few candidate images that are likely to contain the object. We shall call such a mechanism an object *detection* algorithm.

We have implemented such a pruning mechanism, usable when searching for *multi-colored* objects, that relies on non-correspondence matching. It is qualitative in that it uses color histograms of the scene, discarding all spatial information. It is based on the assumption that the histogram of a scene can contain object "signatures" that are invariant over a wide range of scenes and object poses. These signatures are used to detect the presence of multi-colored objects via a fast voting scheme. Although throwing away all spatial information seems rather extreme, the method is suitable for fast object detection.

The object detector mechanism, which produces a "confidence" that the object is in the scene, is run once for each gaze. After it has been run for all the gazes, the confidences are examined to determine which gazes produced "significant" confidences. A simple and effective mechanism for this is the criterion that for a confidence to be significant it must be at least one average deviation greater than the mean confidence for that object over all of the gazes evaluated. By expressing significance in terms of the distance from the mean, we avoid the use of thresholds that may vary with the surroundings or with the specific object detection mechanism being used. The resulting set of significant gazes is then pruned further by eliminating gazes for which an adjacent gaze produced a larger confidence. The gazes remaining in the set after this pruning are those considered most likely to view the

desired object.

6 Qualitative Object Detection via Histograms

In this section we present the object detection mechanism. A "confidence" in the presence of an object in the scene is hypothesized based on the presence or absence of the object's "signature" in a color histogram of the image. Signatures consist of ratios of pairs of buckets in the histogram, and signature detection is accomplished by matching these ratios in the new histogram with a set of stored examples.

6.1 Generating the color histogram

In order to concentrate on recognition, we have chosen to deemphasize the difficult problem of color constancy [Maloney and Wandell, 1986, Hurlbert and Poggio, 1987] by using an opponent-color transform that yields fairly constant color in indoor fluorescent lighting conditions.² This transform takes red (r), green (g), and blue (b) values and produces red-green (O_{rg}) and blue-yellow (O_{by}) values, using the following transform [Lennie and D'Zmura, 1988, Ballard and Brown, 1982]:

$$O_{rg} = r - g$$

$$O_{by} = b - (r + g)/2$$

The two opponent color bands are then histogrammed to form a 16 x 16 two-dimensional red-green vs. blue-yellow histogram. This transform was chosen based on criteria outlined in Kender's [1976] analysis of color transforms [Wixson, 1990b].

6.2 Object signatures

At least one example histogram is stored for every possible target object. We assign a signature to every example histogram h by finding buckets in the histogram for which the number of pixels contained is over some threshold and is greater than the number of pixels contained in any neighboring bucket (*i.e.* is a local maximum). The signature $S(h)$ is simply the set of the ratios of each local maximum to every other local maximum. Thus, if there are m local maxima, the signature consists of $\binom{m}{2}$ ratios. By using the ratios between local maxima, the signatures are scale-independent and consist of those colors that are most likely to be observed in any image containing the object that generated the histogram. Our experiments have shown that for 16 x 16 histograms of everyday multi-colored household objects such as cereal boxes, detergent containers, books, and magazines there are usually ~ 5 local maxima in each histogram.

Given a new histogram h_n , we compute the "goodness of match" between it and an existing histogram h_d from

²This is not to say that color constancy or highlight removal [Klinker *et al.*, 1988, Bajcsy *et al.*, 1989] techniques are not desired, but rather that we are for the moment ignoring this issue. If we had a fast implementation of such methods, we would apply them to the image before performing the opponent-color transform.

the database by averaging a goodness-of-match function for each ratio in the signature of the existing histogram. More formally,

$$goodness(h_n, h_d) = \frac{\sum_{\{a,b\} \in S(h_d)} ratio_match(a, b, h_n, h_d)}{\|S(h_d)\|} \quad (1)$$

where a and b denote buckets in the histogram and

$$ratio_match(a, b, h, h') = weight(\ln r_{a,b}(h) - \ln r_{a,b}(h')) \quad (2)$$

where $weight(x)$ is a very narrow Gaussian with a maximum of 1 centered at 0 and $r_{a,b}(h)$ is simply the ratio of the number of pixels in bucket a to the number of pixels in bucket b in histogram h .

Equation 2, which determines the goodness of match between a pair of buckets in both the new and the database histogram, is the key to the matching procedure. It states that the goodness of match of one pair of buckets is proportional to the distance from unity of the ratio of the ratio of bucket a to bucket b in one histogram to the ratio of a to b in the other histogram. The use of logarithms is simply to make deviations from unity symmetric. The Gaussian is used to assign very close matches a goodness value of 1, while assigning other matches a value close to 0. In all our experiments we used a Gaussian with $\sigma = \ln 1.5$, thereby indicating that a "close match" occurs when the ratios are within a factor of 1.5 of each other.

Using the above goodness calculation, we can compute the confidence that a histogram h_n contains an object o as follows:

$$confidence(h_n, o) = \max_{h_d \in H_o} goodness(h_n, h_d) \quad (3)$$

where H_o is the set of histograms stored in the database as examples of object o .

6.3 Discussion

The idea behind the signature scheme described above is simply that each object has some number of histograms which are stored as examples of that object. This number is usually greater than one since the histogram may change substantially if a different side of the object becomes visible or if substantial lighting changes occur. When testing for the presence of the object in an image, the evaluation of the goodness of match between the image histogram and each example histogram is based only on the most significant buckets in the example histogram. In this way, substantial changes in the background of the scene should not affect the recognition of the object as long as the background pixels do not fall into the buckets used by the example histogram.

The work of Swain [1990] is similar to this in that it is concerned with matching color histograms for object recognition. The key difference between Swain's method, called *histogram intersection*, and the method described above is that histogram intersection is not scale-invariant and thus, without a additional segmentation mechanism, is ill-suited for the object search task.

Finally, it should be noted that Equations 1 and 2 are not the only possible ways of computing a goodness

value. Another possible method might involve using a chi-square test for comparing two binned data sets [Press *et al.*, 1988, pp.489-490] to estimate the similarity between a model signature and the corresponding buckets in a new image.

7 Performance

The object detection method described above has been implemented [Wixson, 1990b].

7.1 Acquiring Example Histograms

The database of example histograms is usually acquired through an iterative process. Our method is to start with a small set of histograms of each object (usually two histograms, taken from two different positions), and run the search task. If an object is missed, we cover the background with black cloth, leaving the object in the same orientation, histogram the scene, and save the scene as an example of the object. By draping the background with black cloth, we eliminate the background signal from the histogram. Leaving the object in the same orientation ensures that the orientation and/or lighting effects that caused the object to be missed originally will be present in the new example histogram. The objects are then moved to different positions, the search task is executed again, and new histograms are learned if necessary. This process is continued until performance is deemed acceptable; the end result is that the database contains ~ 4 example histograms for each object.

7.2 Object Detection

Generation of the 2-D opponent color histogram is performed by a Datacube MaxVideo real-time image processing system, and matching of the new histogram to the example histograms of the target object is performed on a Sun 3/260. A typical search requires that 72 gazes be evaluated. Our system performs this evaluation in 3.5 minutes, taking just under 3 seconds to move to a new gaze, grab the histogram, and compute the match.

Figure 1 shows the direction (but not the distance) of everyday multi-colored objects in a cluttered room in relation to the robot for a typical run of the search task. In addition to these objects, the room contains many other black, gray, or white objects such as tables, cabinets, TV monitors, bookshelves, and chalkboards. Figure 2 shows the gaze directions produced when the search strategy is executed for a "Clorox" detergent box and a "Captain Crunch" cereal box.³ In these figures the area of each circle is proportional to the confidence that the gaze in that direction includes the object. The numbers next to the circles reflect the ordering of the confidences in decreasing order; a circle with number 0 denotes the gaze that the system feels is most likely to contain the object. For each example, we can see that the proper gaze is included in the set of gazes; it is gaze 0 (the highest-ranked gaze) for the "Clorox" box and and

³For these experiments, there were two database histograms for each object, and in the test run, each object was placed at a different position and distance than those at which the example histograms were acquired.

gaze 2 for the "Captain Crunch" box. Results of other experiments can be found in [Wixson and Ballard, 1989, Wixson, 1990b].

In theory, many possible images may give rise to the same histogram and hence to the same signature, resulting in a possibly overwhelming number of false positive matches. Our experiments with the searcher and with making forced choice classifications from a database of images have shown, however, that this does not occur very often. The pruning strategy usually selects less than 7 out of the 72 possible gazes; these are the gazes that have the highest confidence that they contain the desired object. This is more than a 90% reduction in the set of possible gazes. In practice, one of these leftover gazes almost always contains the target object, although this is not always the gaze that produces the maximum goodness. Thus, the false positives that are generated are not numerous enough to cause the correct gaze to be discarded by the pruning strategy.

Situations where the system fails to find the object are almost always due to the presence of objects that massively obscure the signature of the object in the image. In addition, it should be noted that this detection mechanism is not appropriate for objects that exhibit a large amount of specular reflection (for which it is difficult to compute stable example histograms), objects of a single color (for which one cannot compute ratios since only one peak exists in the model histogram), or for objects whose colors are mostly shades of white, gray, or black (since the opponent color transform we currently use maps all of these colors into the same bucket and hence cannot use these as peaks in the histogram). We are working on extensions to handle these cases - Swain [1990] has solved the latter two by using a different color transform and histogramming approach.

8 Conclusion

Introspection reveals that humans perform object search countless times every day. People look for their car keys, for a pay phone, and for their TV's remote control. In addition, everyday tasks such as using a map or following directions require object search in order to find the physical objects which correspond to the landmarks on the map or in the directions. This paper has presented a foundational overview of the object search problem, outlining motivations for the study of object search and identifying three major aspects of the problem. We have only just begun our study of object search; all of the topics described in Section 4 still require a significant amount of research.

Acknowledgments

Thanks go to the Rochester vision group in general, and to Dana Ballard and Michael Swain in particular, for valuable advice and interesting discussions about this paper.

References

- [Ahlswede and Wegener, 1987] Rudolf Ahlswede and Ingo Wegener. *Search Problems*. John Wiley and Sons, 1987.
- [Antes and Penland, 1981] James R. Antes and James G. Penland. Picture context effects on eye movement patterns. In Dennis F. Fisher, Richard A. Monty, and John W. Senders, editors, *Eye Movements: Cognition and Visual Perception*, pages 157-170. Lawrence Erlbaum Associates, 1981.
- [Bajcsy et al., 1989] Ruzena Bajcsy, Sang Wook Lee, and Ales Leonardis. Image segmentation with detection of highlights and inter-reflections using color. In *Image Understanding and Machine Vision, 1989 Technical Digest Series, Vol. 14*, pages 16-19. Optical Society of America, June 1989.
- [Bajcsy, 1988] Ruzena Bajcsy. Active perception. In *Proceedings of the IEEE*, volume 76, pages 996-1005, August 1988.
- [Ballard and Brown, 1982] D.H. Ballard and C.M. Brown. *Computer Vision*. Prentice Hall, Inc., 1982.
- [Ballard, 1989] Dana H. Ballard. Reference frames for animate vision. In *Eleventh International Joint Conference on Artificial Intelligence*, pages 1635-1641, August 1989.
- [Biederman et al., 1973] Irving Biederman, Arnold L. Glass, and E. Webb Stacy, Jr. Searching for objects in real-world scenes. *Journal of Experimental Psychology*, 97(1):22-27, 1973.
- [Bolle et al., 1989] Ruud M. Bolle, Andrea Califano, and Rick Kjeldsen. Data and model driven foveation. I.b.m. research report, Exploratory Computer Vision Group, IBM T.J. Watson Research Center, 1989.
- [Brooks, 1987] Rodney A. Brooks. Visual map making for a mobile robot. In Martin A. Fischler and Oscar Firschein, editors, *Readings in Computer Vision*, pages 438-443. Morgan Kaufmann, 1987.
- [Brown, 1988] Christopher M. Brown. The rochester robot. Technical Report 257, University of Rochester Computer Science Dept., August 1988.
- [Cohen, 1981] Karen M. Cohen. The development of strategies of visual search. In Dennis F. Fisher, Richard A. Monty, and John W. Senders, editors, *Eye Movements: Cognition and Visual Perception*, pages 271-288. Lawrence Erlbaum Associates, 1981.
- [Connell, 1989] Jonathan H. Connell. *A Colony Architecture for an Artificial Creature*. PhD thesis, M.I.T., 1989.
- [Feldman and Sproull, 1977] Jerome Feldman and Robert Sproull. Decision theory and artificial intelligence II: The hungry monkey. *Cognitive Science*, 1:158-192, 1977.
- [Fisher et al., 1981] Dennis F. Fisher, Richard A. Monty, and John W. Senders, editors. *Eye Movements: Cognition and Visual Perception*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1981.

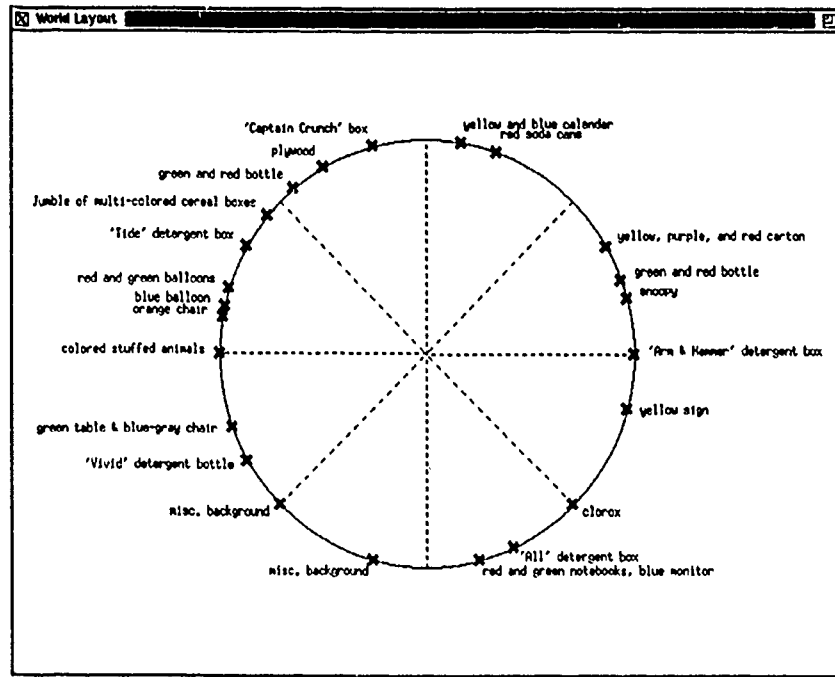


Figure 1. Top-view of the laboratory environment for a typical test run showing the direction (but not the distance) of each object with respect to the robot. Dashed lines are for easy reference between the figures.

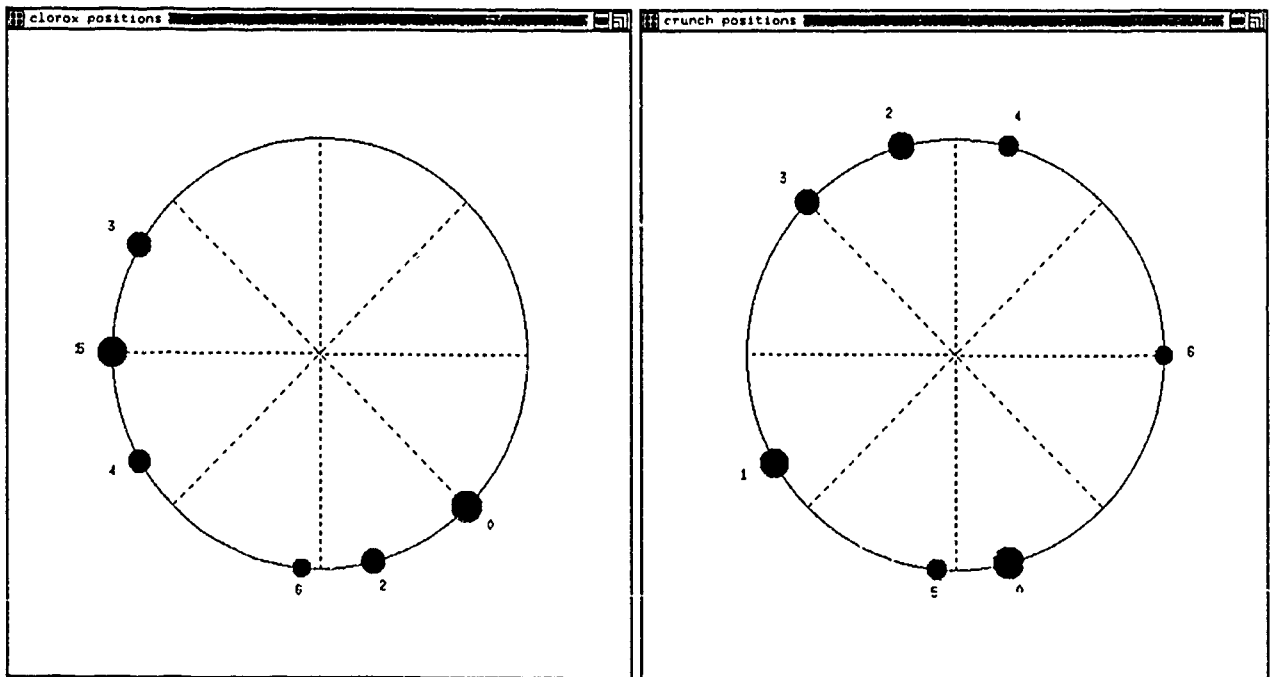


Figure 2. Gaze directions produced by the object search mechanism for the "Clorox" detergent box and the "Captain Crunch" cereal box. Area of circle is proportional to the confidence in that gaze. Numbers next to circles reflect the ordering of the confidences in decreasing order. Dashed lines are for easy reference between the figures.

- [Garvey, 1976] Thomas D. Garvey. Perceptual strategies for purposive vision. Technical Note 117, SRI International, September 1976.
- [Goad, 1987] Chris Goad. Special purpose automatic programming for 3d model-based vision. In Martin A. Fischler and Oscar Firschein, editors, *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 371-381. Morgan Kaufmann Publishers, Inc., 1987.
- [Hurlbert and Poggio, 1987] Anya C. Hurlbert and Tomaso A. Poggio. Learning a color algorithm from examples. In *Neural Information Processing Systems*, pages 622-631, 1987.
- [Huttenlocher, 1988] Daniel P. Huttenlocher. Three-dimensional recognition of solid objects from a two-dimensional image. Technical Report 1045, MIT AI Lab, 1988.
- [Huttenlocher, 1989] Daniel P. Huttenlocher. personal communication, November 1989.
- [Kender, 1976] John R. Kender. Saturation, hue, and normalized color: Calculation, digitization effects, and use. Technical report, Carnegie-Mellon University, November 1976.
- [Klinker et al., 1988] Gudrun J. Klinker, Steven A. Shafer, and Takeo Kanade. Image segmentation and reflection analysis through color. In *Proceedings of the DARPA IUS Workshop*, pages 838-853, 1988.
- [Krotkov, 1987] Eric Paul Krotkov. Exploratory visual sensing for determining spatial layout with an agile stereo camera system. Technical Report MS-CIS-87-29, University of Pennsylvania Dept. of Computer and Information Science, April 1987.
- [Lamdan and Wolfson, 1988] Yehezkel Lamdan and Haim J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. Technical Report Technical Report 368, New York University Computer Science Dept., May 1988.
- [Lennie and D'Zmura, 1988] Peter Lennie and Michael D'Zmura. Mechanisms of color vision. *CRC Critical Reviews in Neurobiology*, 3(4):333-400, 1988.
- [Lowe, 1985] David Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, 1985.
- [Lowe, 1989] David G. Lowe. Fitting parameterized 3-d models to images. Technical Report 89-26, University of British Columbia Computer Science Dept., December 1989.
- [Maloney and Wandell, 1986] Laurence T. Maloney and Brian A. Wandell. Color constancy: a method for recovering surface spectral reflectance. *Journal of the Optical Society of America A (JOSA-A)*, 3(1):29-33, January 1986.
- [McIvor, 1988] Alan M. McIvor. An analysis of lowe's model-based vision system. In *Proceedings of the Fourth Alvey Vision Conference*, August 1988.
- [Morris and Horne, 1959] Ailene Morris and E. Porter Horne, editors. *Visual Search Techniques*, Washington, D.C., 1959. Armed Forces - NRC Committee on Vision, National Academy of Sciences - National Research Council.
- [Press et al., 1988] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, 1988.
- [Rabbitt, 1978] P. Rabbitt. Sorting, categorization, and visual search. In E. Carterette and M. Friedman, editors, *The Handbook of Perception: Perceptual processing*, Vol. IX. Academic Press, 1978.
- [Rimey and Brown, 1990] Ray Rimey and Christopher M. Brown. Sequential behavior as a selective attention mechanism: modelling eye movements with hidden markov models. Technical report, Computer Science Dept., University of Rochester, 1990.
- [Stark and Ellis, 1981] Lawrence Stark and Stephen R. Ellis. Scanpaths revisited: Cognitive models direct active looking. In Dennis F. Fisher, Richard A. Monty, and John W. Senders, editors, *Eye Movements: Cognition and Visual Perception*, pages 193-226. Lawrence Erlbaum Associates, 1981.
- [Swain et al., 1989] Michael J. Swain, Lambert E. Wixson, and Dana H. Ballard. Object identification and search: Animate vision alternatives to image interpretation. In *Proceedings of the NATO Advanced Research Workshop on Robots and Biological Systems*, June 1989.
- [Swain, 1990] Michael J. Swain. *Recognizing Colored Objects*. PhD thesis, University of Rochester Computer Science Dept., 1990. In preparation.
- [Treisman, 1988] Anne Treisman. Features and objects: The fourteenth bartlett memorial lecture. *The Quarterly Journal of Experimental Psychology*, 40(2):201-237, 1988.
- [Visual Search, 1973] Committee on Vision, Division of Behavioral Sciences, National Research Council. *Visual Search*, Washington, D.C., 1973. National Academy of Sciences.
- [Wixson and Ballard, 1989] Lambert E. Wixson and Dana H. Ballard. Real-time detection of multi-colored objects. In *SPIE Sensor Fusion II: Human and Machine Strategies*, volume 1198, November 1989.
- [Wixson, 1990a] Lambert E. Wixson. The acquisition and utilization of high-level knowledge about spatial relationships to search for an object. Technical report, University of Rochester Computer Science Department, April 1990.
- [Wixson, 1990b] Lambert E. Wixson. Object search. A pervasive (but unstudied) visual task. Technical report, University of Rochester Computer Science Department, April 1990.
- [Yarbus, 1967] A.L. Yarbus. *Eye Movements and Vision*. Plenum Press, 1967.

PERCEPTUAL ORGANIZATION OF OCCLUDING CONTOURS

Lance R. Williams

Computer and Information Science Department
University of Massachusetts at Amherst
Amherst, Mass., USA *

Abstract

Contours corresponding to surface boundaries are readily perceived by human observers even when local evidence in the form of measurable image brightness gradients is completely absent. Figural completion and illusory contours offer valuable clues to the computational processes employed by the human visual system in constructing image contours from the incomplete and fragmentary evidence provided by image brightness changes. In particular, they suggest that whether or not a gap in an image contour is completed depends on a non-local process with specific knowledge of surfaces and occlusion. In my work, the mechanics of occlusion of one surface by another are described by a set of integer linear constraints. These constraints insure that the output of a contour grouping process is physically valid and consistent with the image evidence. Among the many feasible solutions, the most compelling is the solution which best explains the presence and form of image structure. The problem of computing a complete and consistent surface boundary representation is reduced to solving an integer linear program.

1 Introduction

Marr cited the failure of edge detection and intensity based segmentation as evidence that the goal of low-level vision (as articulated to that point) was ill-specified. He argued that a more appropriate goal for low-level vision is the computation of a representation he called the $2\frac{1}{2}D$ Sketch [15]. The $2\frac{1}{2}D$ Sketch is an explicit representation of the orientation and depth of visible surfaces in retinocentric coordinates. Importantly, discontinuities in orientation and depth are also explicitly represented. A great deal of effort has been expended during the last ten years with the goal of computing representations of visible surfaces by diverse methods including shape from shading, texture, stereo, general sensor motion, etc.

Witkin and Tenenbaum [27] emphasized the importance of *perceptual organization* as a goal for low-level

vision. In their view, perceptual organization provides the precursors of higher-level (i.e. more specialized) representations by detecting and explicitly representing image structure. They maintain that local quantitative processes are incapable of reliably computing the $2\frac{1}{2}D$ Sketch, and that additional constraints provided by non-local grouping processes are required. Compared to the effort devoted to specific "shape-from" processes, there has been relatively little work aimed at understanding the role which image contours play in constraining scene structure (exceptions include [1,2,14,19]). Image contours corresponding to discontinuities in depth are called *occluding contours*. Many natural scenes (e.g. rainforests, desktops) achieve much of their complexity through clutter and are dominated by discontinuities in depth. Occluding contours are an important source of constraints, defining the domains within which individual "shape-from" processes operate. Accordingly, understanding the grouping processes which create occluding contours is an important problem in computer vision.

Recent years have seen progress in identifying and describing image contours (occluding and otherwise). This is especially true of straight lines [3,7,17]. The particular assumptions that different grouping methods make, and the degree and kinds of evidence each considers sufficient, vary widely. None specifically considers the effects of occlusion and interference due to clutter. The Burns algorithm [7] adopts a conservative strategy. Its grouping criterion is similarity of brightness gradient direction within a (fixed) neighborhood around a particular edge pixel. Although this criterion often results in fragmented contours containing large gaps (i.e. *undergrouping*), it rarely produces inappropriate lines (i.e. *overgrouping*). The Boldt zero crossing grouping algorithm [3] is at the other extreme. Its notions of proximity and straightness are coupled to a scale parameter that gradually increases as the algorithm runs (i.e. Boldt's algorithm is *non-local*). As a result, Boldt's algorithm is capable of detecting and creating straight lines even when the image evidence at pixel scale (and at larger scales) is fragmented and of low contrast. This is a very powerful capability, but results in a larger number of overgrouping

*This research was supported by the Defense Advanced Research Projects Agency under RADC contract F30602-87-C-0140 and Army ETL contract DACA76-89-C-0017.

errors. Because its reasoning is (essentially) based upon straightness alone, Boldt's algorithm simply doesn't have enough information to make the "correct" grouping decisions all of the time. As with edge detection, part of the problem is the lack of any reasonable definition of "correct"; Witkin and Tenenbaum [27] ask "Is there any objective sense in which some groupings are true and accurate, while others are false or inaccurate?"

Visual psychologists classify image contours as either *modal* or *amodal*, depending on whether or not there is a subjective impression of brightness change across the contour [13]. For example, an image contour which continues behind an opaque surface is amodal within the interval where it is blocked from view; this is referred to as *amodal completion*. More surprising is the fact that under appropriate circumstances the human visual system constructs image contours that are subjectively experienced as changes in *apparent brightness* even though there is no measurable brightness gradient. These are called *illusory contours*¹. Both amodal completion and illusory contours offer valuable clues to the computational processes employed by the human visual system in constructing image contours from the incomplete and fragmentary evidence provided by local image brightness changes. The problem of computing the *shape* of an illusory contour joining two boundary fragments was originally studied by Ullman [25] (see [10] for a more recent treatment). But the problem of exactly *when* an illusory contour should join two boundary fragments is yet to be answered in precise computational terms (as [15,27] and others have pointed out). This is not to imply that the problem has gone unstudied. Indeed, visual psychologists have documented many of the conditions required to induce illusory contours in human vision [18]. This is epitomized by the carefully designed figures of Kanizsa [13] which are the most dramatic illustrations of the phenomenon (Figure 1). Finally, the computational model outlined in this paper is inspired in part by Irvin Rock's view of human perception as problem solving [20].

Although a representation like the $2\frac{1}{2}$ D Sketch is not necessary for all vision tasks (e.g. model based object recognition), it is clearly required for others (e.g. visual obstacle avoidance and navigation in unmodeled environments). Yet robust computation of the $2\frac{1}{2}$ D Sketch under the most general conditions (i.e. where all surfaces aren't densely textured and smooth) is probably impossible without additional constraints derived from

¹Because illusory contours are often referred to as *optical illusions*, there is sometimes a tendency to dismiss them as peculiarities of human vision which occur only under highly contrived conditions. An alternate view is that figures like Kanizsa's Triangle reveal the incongruity between real and apparent brightness more dramatically than is typical but that the mechanism responsible for illusory contours is actually very general.

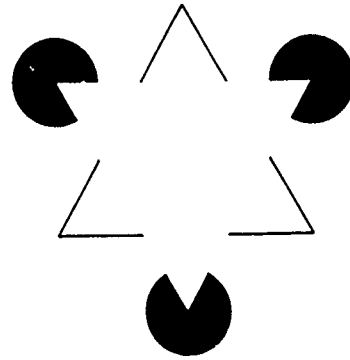


Figure 1: Kanizsa's Triangle.

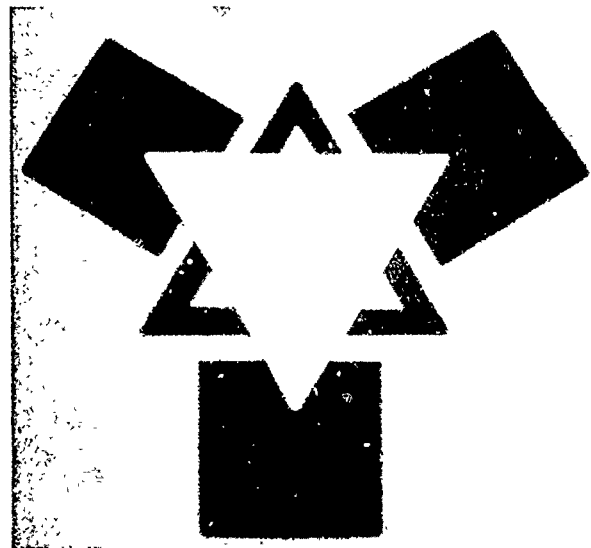


Figure 2: A Colorforms version of the Kanizsa Triangle.

occluding contours (which are the products of grouping processes). Happily, the semantics of the $2\frac{1}{2}$ D Sketch can provide objective grouping criteria. Whether the human visual system is willing or unwilling to complete a gap in an image contour is determined by a non-local process with specific knowledge of the mechanics of surfaces and occlusion.

2 Demonstration System

The demonstration system is a working model of a "complete" perceptual organization system, capable of producing a surface boundary representation from figural cues alone (albeit within a highly restricted domain). As a test domain, we chose simple scenes built from flat vinyl cutout surfaces of uniform reflectance called Colorforms. Colorforms are straight sided, but flexible, and it is possible to build fairly complex scenes involving oc-

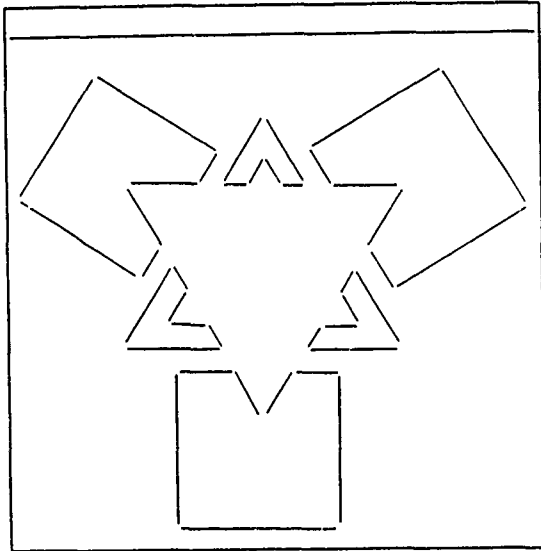


Figure 3: Image lines for the Kanizsa Triangle.

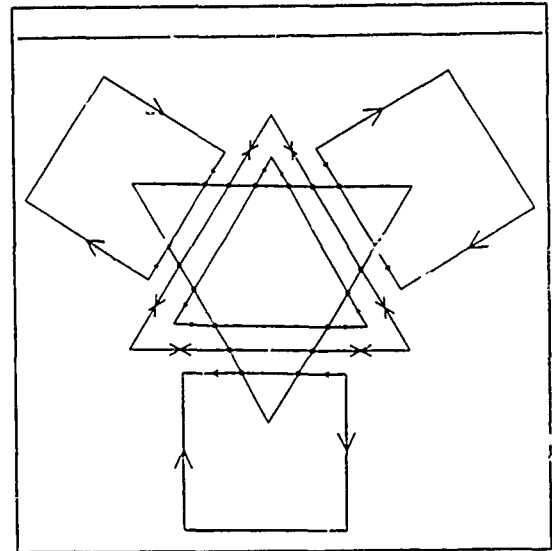


Figure 4: Contour graph for the Kanizsa Triangle

clusion, holes and interwoven surfaces. The current system is designed to complete gaps in the straight sections of occluding contours, but isn't able to cope with more complex omissions such as missing corners or missing sides. Nevertheless, even with this restriction, it is able to solve relatively complex perceptual problems such as the construction of an illusory triangle in a Colorforms equivalent of the Kanizsa Triangle (Figure 2). The system operates in two stages: 1) A problem posing stage; and 2) A problem solving stage.

2.1 From Contours to Surface Boundaries

In the problem posing stage image evidence is collected and incorporated in a graph, called the *contour graph*. The contour graph is an explicit representation of *primitive* image structure[27] and corresponds approximately to Marr's *full primal sketch*[15]. It is composed of two types of vertices and three types of edges. Every vertex is located at a point in the image and every edge is a contour joining two vertices. The initial edges of the contour graph are called *image lines*, and are created² by Boldt's zero crossing grouping algorithm [3] (Figure 3).

Image lines are contours with a measurable image brightness gradient. Each image line joins its two *end-points*, which are the initial vertex type. Proximal end-points of image lines satisfying certain other simple criteria are joined with a second edge type called a *corner*. Next, all pairs of roughly collinear image lines (as determined by the mean square error of a line fit to the four endpoints) are identified. The near endpoints of each such pair are joined by a third edge type, the *virtual line*

[13,15,24]. A virtual line (for which there is generally no corresponding image brightness gradient) is a token which serves as an explicit representation of a collinear image line pair. Finally, wherever a virtual line intersects another virtual line, or a virtual line intersects an image line, the two lines are split into four sub-segments and joined by a new type of vertex, called a *crossing*. This insures that the graph remains planar. The contour graph computed for the Kanizsa Triangle is depicted³ in Figure 4.

By writing a fixed number of linear constraints for each vertex and edge in the contour graph, an integer linear program is generated. During the problem solving stage, branch and bound search is used to find its optimal feasible solution. The optimal feasible solution defines the *boundary graph*, which is a labeled sub-graph. This is consistent with Witkin and Tenenbaum's [27] claim that "naively perceived structure survives more or less intact when a semantic context is established... the difference between naive and informed perception amounting to little more than labeling the perceptual primitives." The edges of the boundary graph are labeled with a sign of occlusion and a depth index (hidden lines are displayed dashed). The boundary graph corresponding to the optimal feasible solution of the integer linear program is depicted in Figure 5. An alternate organization, which is a feasible but non-optimal solution, appears in Figure 6.

2.2 Physical Validity

We now demonstrate that all constraints necessary to insure the physical validity of the boundary graph are expressible in an integer linear program. The first constraint enforced is that every occluding contour must

²In the case of the Colorforms version of the Kanizsa Triangle, all evidence of the center triangle was first removed by filtering the initial zero crossing segments on gradient magnitude.

³The arrows indicate the figure-ground sense of bar tokens (See Section 2.4).

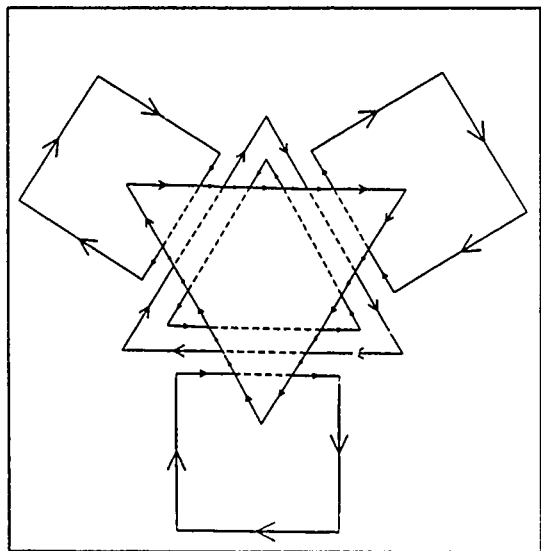


Figure 5: The optimal feasible solution.

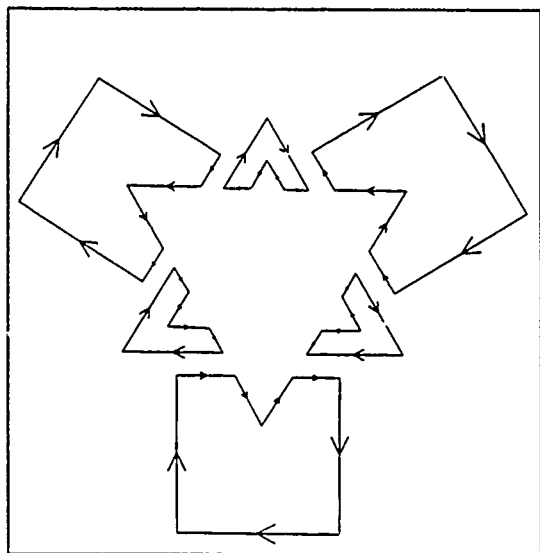


Figure 6: A feasible solution which is non-optimal.

have one of two signs of occlusion (i.e. \rightarrow and \leftarrow). The two possible signs of occlusion of image line i are represented as 0-1 valued integers x_i and x'_i . Image line i is an occluding contour when either $x_i = 1$ or $x'_i = 1$, otherwise $x_i = x'_i = 0$ and i is a non-occluding contour. Using this representation, the necessary constraint is the following linear inequality:

$$x_i + x'_i < 1 \quad (1)$$

Since the projections of complete surface boundaries are closed contours, all occluding contours must be part of cycles in the surface boundary graph. The continuation of an image line through more than one virtual line or corner at each endpoint is also prohibited.

Furthermore, no virtual line or corner can act as an occluding contour without its "sponsoring" image line,

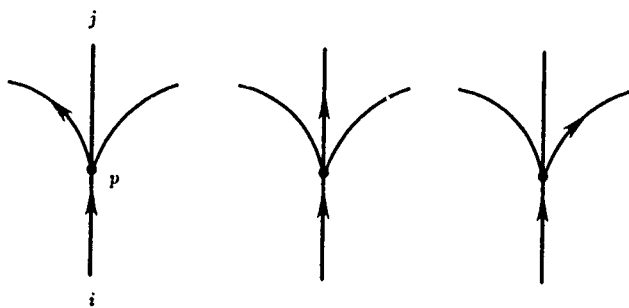


Figure 7: Potential continuations, j , of image line i through endpoint p .

virtual lines and corners only join image line fragments in the process of constructing surface boundaries. Finally, the sign of occlusion of the continuation must be unique and consistent with the sign of occlusion of the image line. This insures that every cycle in the surface boundary graph has a single sign of occlusion (i.e. either clockwise or counter-clockwise in direction). Let j be the virtual lines and corners which are potential continuations of image contour i at endpoint p (Figure 7). Two constraints per endpoint guarantee all of the above:

$$x_i = \sum_j x_j \quad (2)$$

$$x'_i = \sum_j x'_j \quad (3)$$

These constraints play a role similar to the conservation constraints in a network flow problem. In this case, sign of occlusion is conserved at each endpoint. The right side of each inequality is the sum of all continuations with sign of occlusion consistent with the sign of the sponsoring image line on the left side. Since the left sides are 0-1 valued, the right sides are likewise bounded and at most one virtual line or corner can serve as a continuation. When $x_i = x'_i = 0$, the right sides of both inequalities must also equal zero, which insures that no virtual line or corner can become an occluding contour independently of its sponsoring image line. Conversely, image line i can not become an occluding contour without continuation through endpoint p , prohibiting "dangling" endpoints. This guarantees that all occluding contours are part of cycles.

We now systematically present the linear constraints required to simulate the more complicated behavior associated with the occlusion of one opaque surface by another. Recall that as part of the process of constructing the contour graph, at every point where one contour crosses another, the contours are split into four new con-

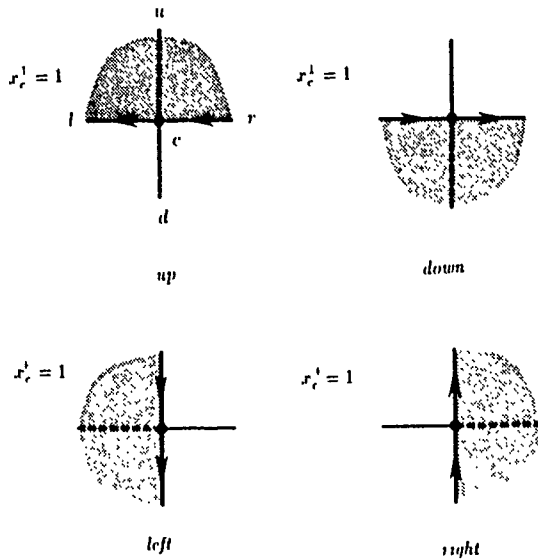


Figure 8: The four principal crossing states.

tours and joined by a *crossing* node. Call the four contours u , d , l and r and the crossing node c (Figure 8). Associated with each of the four contours are 0-1 valued integers x and x' representing their signs of occlusion. Also associated with each segment is a positive integer variable n representing the relative depth of the contour (i.e. the number of surfaces between the contour and the eye or camera). Certain constraints are immediately apparent. First of all, the signs of occlusion of contour u and contour d must be consistent. Likewise for contour l and contour r . As simple equality constraints, they can be enforced by substitution and needn't actually appear in the linear program: $x_u = x_d$, $x'_u = x'_d$, $x_l = x_r$ and $x'_l = x'_r$.

A second observation is that if u and l (and by implication d and r) are occluding contours, then the surface which u bounds (call it S_u) is either above or below the surface which l bounds (call it S_l). This is independent of the specific signs of occlusion of u or l . When one considers that only the sign of occlusion of the uppermost surface has any effect on the relative depths of the four contours (i.e. n_u , n_d , n_l and n_r), it becomes clear that crossing c can be in one of four principal states. The specific state is determined by which of u , d , l or r is being occluded by the uppermost surface. When S_l is above S_u , and the sign of occlusion of l is $r \rightarrow l$ (i.e. $x'_l = 1$), then the crossing is in the *up* state (denoted by \uparrow). If contour l 's sign of occlusion is $l \rightarrow r$ (i.e. $x_l = 1$) then the crossing is in the *down* state (denoted by \downarrow). When S_u is above S_l the crossing is either in the *left* (\leftarrow) state or the *right* (\rightarrow) state, depending on whether the sign of occlusion of u is $u \rightarrow d$ (i.e. $x'_u = 1$) or $d \rightarrow u$ (i.e. $x_u = 1$). The four states are represented in the linear program with four 0-1 valued variables x_c^\uparrow , x_c^\downarrow , x_c^\leftarrow and

x_c^\rightarrow . Crossing c is in the *up* state exactly when $x_c^\uparrow = 1$ and $x_c^\downarrow = x_c^\leftarrow = x_c^\rightarrow = 0$. The other three states are represented similarly. Having established a representation, it is now possible to describe the first constraint enforced at every crossing. It requires the crossing to be in one of the four states when both u and l are occluding contours:

$$x_u + x'_u + x_l + x'_l \leq x_c^\uparrow + x_c^\downarrow + x_c^\leftarrow + x_c^\rightarrow + 1 \quad (4)$$

When u and l are occluding contours, the left side of the inequality equals two, forcing at least one of x_c^\uparrow , x_c^\downarrow , x_c^\leftarrow and x_c^\rightarrow to equal one. Another constraint makes the four states mutually exclusive:

$$x_c^\uparrow + x_c^\downarrow + x_c^\leftarrow + x_c^\rightarrow \leq 1 \quad (5)$$

The specific signs of occlusion which are preconditions for each of the four states appear on the right sides of the inequalities which follow:

$$x_c^\uparrow \leq x'_l \quad (6)$$

$$x_c^\downarrow \leq x_l \quad (7)$$

$$x_c^\leftarrow \leq x'_u \quad (8)$$

$$x_c^\rightarrow \leq x_u \quad (9)$$

For example, crossing c can only be in the *left* state (i.e. $x_c^\leftarrow = 1$) when contour u 's sign of occlusion is $u \rightarrow d$ (i.e. $x'_u = 1$).

It is important to note that the four principal crossing states stand for specific differences in relative depth across the crossing node. For a particular surface boundary graph to be feasible, the sum of the depth index differences around every cycle must equal zero. The following constraints serve to define the crossing states as relative depths:

$$n_u - n_d = x_c^\uparrow - x_c^\downarrow \quad (10)$$

$$n_l - n_r = x_c^\leftarrow - x_c^\rightarrow \quad (11)$$

2.3 Fidelity to Image Data

Physical validity is a necessary but not a sufficient condition for feasibility of the surface boundary graph. For a solution to be feasible, it must be both physically valid and consistent with the image data⁴. First and foremost, image lines are, by their very nature, visible. We therefore require that their relative depth indices in the surface boundary graph equal zero (i.e. for every image line i , $n_i = 0$). This can be enforced by simply excluding all n_i from the linear program, and needn't increase the size of the constraint matrix.

⁴As Irvin Rock points out[22] "the solution must conform to the proximal stimulus."

Equally important, if a virtual line is an occluding contour, and its depth index is equal to zero (indicating that it should be visible) then its absence as an image line should be explainable⁵. Consider virtual line j which joins image lines l and r (possibly through an arbitrary number of additional contours and crossings; see Figure 9). Let x_j be the sign of occlusion of contour j corresponding to the $l \rightarrow r$ direction and x'_j be the opposite sign. Depending on its sign of occlusion, contour j bounds either surface S_j or S'_j . Associated with S_j is reflectance φ_j and with S'_j is reflectance φ'_j . Assuming roughly uniform illumination and reflectance, φ_j and φ'_j can be approximated by the average brightness within narrow regions on either side of contour j . In a similar manner, we can compute φ_l , φ'_l , φ_r and φ'_r associated with surfaces S_l , S'_l , S_r and S'_r .

What conditions, at minimum, should exist before a visual system constructs an illusory contour? Are there any circumstances under which a surface boundary projects to the image plane with no appreciable brightness gradient? Let $|\Delta\varphi_j|$ be the magnitude of the brightness gradient across contour j , then:

$$|\Delta\varphi_j| = |\varphi_j - \varphi'_j| \quad (12)$$

When $x_j = 1$ and $n_j = 0$ then j joins l and r to form a visible occluding contour bounding a common surface: $S_l \equiv S_j \equiv S_r$. Assuming that the surface has roughly constant reflectance, we conclude that $\varphi_l \approx \varphi_j \approx \varphi_r$. Similarly, if $x'_j = 1$ and $n_j = 0$, then j joins l and r to form a visible section of occluding contour bounding surface $S'_l \equiv S'_j \equiv S'_r$. We conclude that $\varphi'_l \approx \varphi'_j \approx \varphi'_r$. Since $n_j = 0$, there are no surfaces between the boundary and the eye, and there will usually be a detectable contour in the image. However, when a surface occludes another surface with similar reflectance then $\varphi_j \approx \varphi'_j$ and $|\Delta\varphi_j| \approx 0$. This suggests that illusory contours should be permitted only where the available measurements of image brightness are consistent with the case of two overlapping surfaces of roughly constant and approximately equal reflectance. This effect can be achieved by adding Constraint 13 unless $\varphi'_j \approx \varphi_l \approx \varphi_j \approx \varphi_r$ and adding Constraint 14 unless $\varphi_j \approx \varphi'_l \approx \varphi'_j \approx \varphi'_r$:

$$x_j \leq n_j \quad (13)$$

$$x'_j \leq n_j \quad (14)$$

2.4 Preference

The constraint matrix defines the feasible region of the linear program. All integer solution vectors within the feasible region correspond to physically valid boundary graphs with boundary depth indices consistent with the

⁵This requirement is waived for very short lines (i.e. lines with length less than 2 pixels).

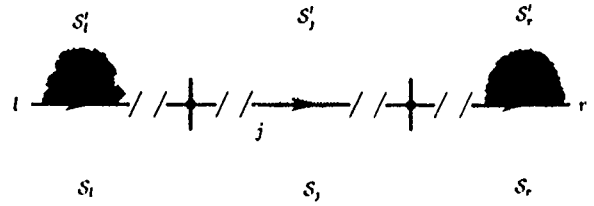


Figure 9: Virtual line j joins images lines l and r .

visibility (or non-visibility) of the image lines. Not all solution vectors are equally compelling however. For example, the zero vector is always within the feasible region, and corresponds to a boundary graph with no surface boundaries; all image lines are unrelated non-occluding contours. Of course, this interpretation completely ignores all figural evidence to the contrary, and is generally not the most compelling. What basis exists for preferring one feasible solution over another? What are the coefficients of the cost vector which defines the objective function?

An important factor in human vision (often mentioned in connection with the Kanizsa Triangle) is the tendency towards figural completion. Since all occluding contours must be part of cycles in the boundary graph, a preference for complete figures can be achieved by maximizing the number of occluding contours. Of course occluding contours can be embedded in boundary graph cycles only through liberal use of virtual lines and corners, and among the mechanisms of completion, virtual lines are preferred. With respect to human vision, this can be justified by invoking the Gestalt law of good continuation. Recall that virtual lines are used to join collinear image lines while corners join image lines with proximal endpoints. Where as collinearity of two image lines almost certainly implies an underlying *common cause* [20], endpoint proximity is frequently an artifact of occlusion of one surface by another. Consider the fact that the brightness gradient magnitude measured along an occluding contour varies at points where the surface it bounds crosses other surfaces of different reflectance. This results in a plague of "false" corners. In contrast, a virtual line which can not be incorporated into the boundary graph requires crediting chance alone as explanation for the collinearity of the sponsoring image lines. According to Rock [20], "The perceptual system detects continuity of direction among contours. Once doing so, not to accept two or more elements as parts of one larger entity is to accept that continuity as the result of coincidental placement in space of these elements, that is, of elements that have no intrinsic relationship to one another." It follows that boundary graphs which make

greater use of virtual lines are more compelling because their acceptance requires making fewer assumptions of accidental figural alignment.

Two preference factors have been discussed so far: A preference for complete figures in general; and a preference for complete figures incorporating virtual lines as opposed to corners. Both indirectly determine the sign of occlusion of those contours which act as facilitators (through crossing vertices) of figural completion. An as yet unaddressed factor in preference concerns the determination of a contour's sign of occlusion in the many ambiguous cases where it is not directly constrained by a role in figural completion. In fact, it might be legitimately asked whether or not asserting the existence of an occluding contour is appropriate in such underconstrained cases. Yet it is well documented that in similarly ambiguous situations, the human visual system displays strong and characteristic *figure-ground* preferences. Among the most important of these is: A preference for convex figure and concave ground [20]. This is easily implemented in the integer linear program by assigning different costs to the two possible signs of occlusion associated with each corner. Corner signs of occlusion consistent with convex figure are rewarded while those consistent with concave figure are penalized. Together with the preference for closed figure, this has the effect of increasing the number of clockwise cycles (i.e. solids) in the boundary graph at the expense of the counter-clockwise cycles (i.e. holes). The ultimate physical basis for this preference may be the asymmetric roles played by figure and ground in the physical world; the space occupied by figure is opaque, while the complementary space is transparent⁶.

Another set of preferences are associated with the detection and explanation of symmetries of various kinds. The simplest of these is: A preference for perceiving the space between closely spaced parallel lines as figure [20]. As part of the process of creating the contour graph, all pairs of parallel lines with opposite brightness gradient and with significant mutual overlap are identified as *bar* tokens. A bar is *instantiated* when the image lines which bound it are assigned the signs of occlusion consistent with the bar being figure. The preferred interpretation reconciles all other competing factors in figure-ground preference with the goal of instantiating the largest number of bars. Let b be a bar token bounded on the left and right by image lines l and r . Let x_l and x_r be the signs of occlusion consistent with the bar being figure and x_b be a new 0-1 valued variable representing the instantiated bar. The following linear constraints insure that b

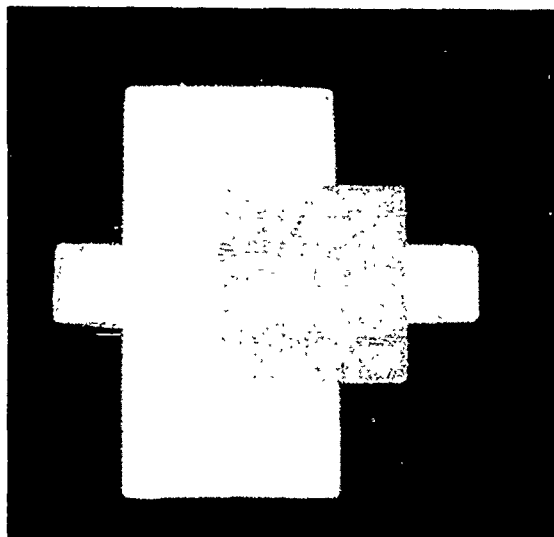


Figure 10: Blocks scene.

will be instantiated (i.e. $x_b = 1$) only when $x_l = 1$ and $x_r = 1$:

$$x_b \leq x_l \quad (15)$$

$$x_b \leq x_r \quad (16)$$

We are finally ready to define an objective function that is sufficient for the Colorforms domain. Let i , v , c and b range over the sets of image lines, virtual lines, corners and bars respectively. Maximize the following:

$$\begin{aligned} & \alpha \sum_i (x_i + x'_i) + \beta \sum_v (x_v + x'_v) \\ & + \gamma \sum_c (x_c - x'_c) + \delta \sum_b x_b \end{aligned}$$

Each of the four terms in the objective function reflects one of the four preference factors discussed so far. A large α results in a heavy bias for interpreting image lines as occluding contours (and embedding them in closed boundary graph cycles). The relative preference for the two mechanisms of completion, virtual lines and corners, depends on β and γ . The preference for convex figure and concave ground is implemented by means of a minus sign in the γ term, which rewards the former and penalizes the latter. The value of δ determines the relative importance of incorporating the figure-ground sense which each of the bars requires to be instantiated. The actual values of α , β , γ and δ were chosen experimentally (with the goal that the system's results should match the solutions preferred by the human visual system). All of the results in this paper were obtained with $\alpha = 5$, $\beta = 2$, $\gamma = 1$ and $\delta = 2$. Deriving values for these parameters from first principles (or through psychophysical experimentation) is an open problem.

⁶Hoffman and Richards [9] suggest that the human visual system segments contours into parts at negative minima of curvature. If this is true, then a preference for convex figure and concave ground leads naturally to a contour description with a fewer number of parts.

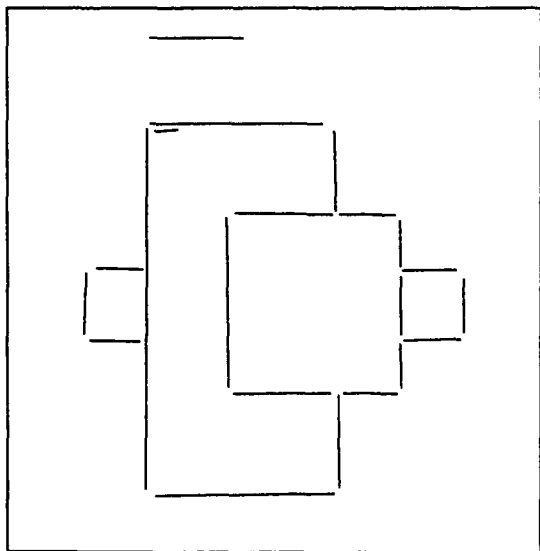


Figure 11: Image lines for blocks.

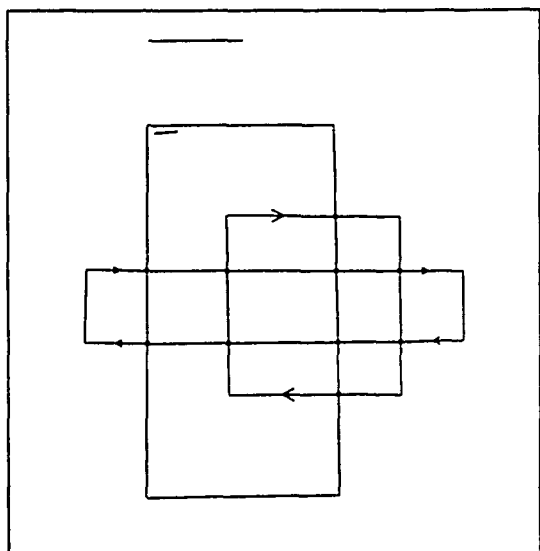


Figure 12: Contour graph for blocks.

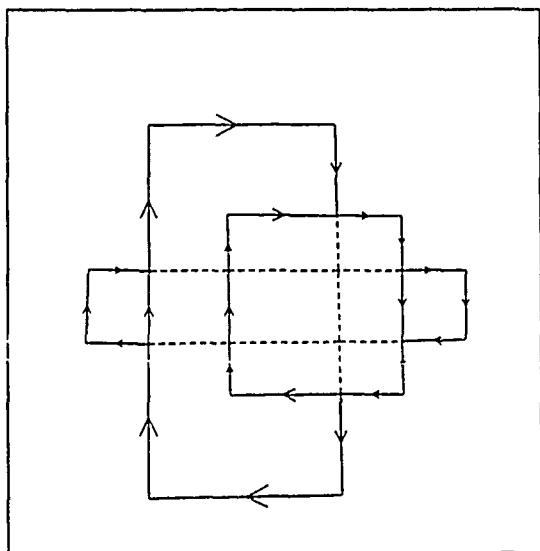


Figure 13: Boundary graph for blocks.

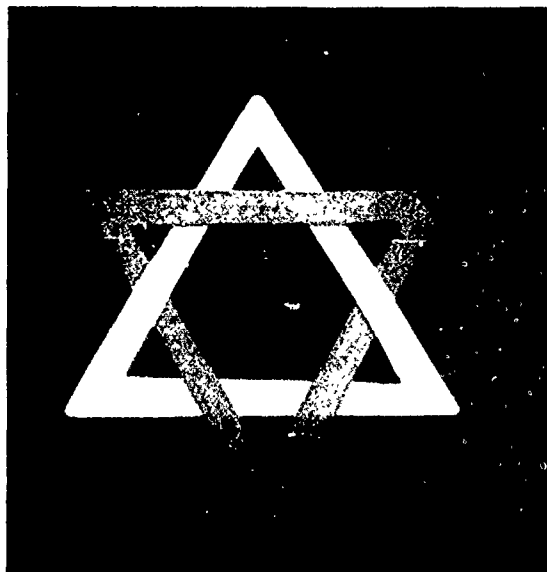


Figure 14: Star scene.

3 Additional Results

Figure 10 is a digitized image of a simple Colorforms scene composed of two rectangles and a square. These were arranged so that local evidence for the occlusion of the horizontally oriented rectangle by the larger rectangle is hidden by the square. Figure 11 shows the image lines produced by Boldt's algorithm. The contour graph, computed from the image lines, and showing the addition of virtual lines, corners and crossings, appears in Figure 12. The arrows on the image lines indicate the figure-ground sense individual bar tokens require to be instantiated. Finally, Figure 13 is the boundary graph corresponding to the optimal feasible solution of the integer linear program. The relative depths of the contour segments under the square, although locally ambiguous, are assigned the values required to achieve consistency of the figure as a whole.

The human visual system is unwilling to complete a gap in a contour unless there is evidence of occlusion. A similar effect can be demonstrated with two meshed triangles (Figure 14). Figure 15a shows the image lines produced by Boldt's algorithm. In Figure 15b, the image lines corresponding to one of the triangles have been artificially removed. The contour graphs computed from the two sets of image lines appear in Figures 16a and 16b. Examination of Figure 17a shows that the demonstration system readily completes both triangles in the case of the intact figure, but the boundary graph in Figure 17b contains only unrelated fragments. The collinearity among the fragments is explicitly represented at the level of primitive structure through virtual lines, but the virtual lines are not interpreted as surface boundaries.

Construction and representation of the contour graph was greatly facilitated by the use of a Lisp based database called the ISR [6]. The ISR was specifically

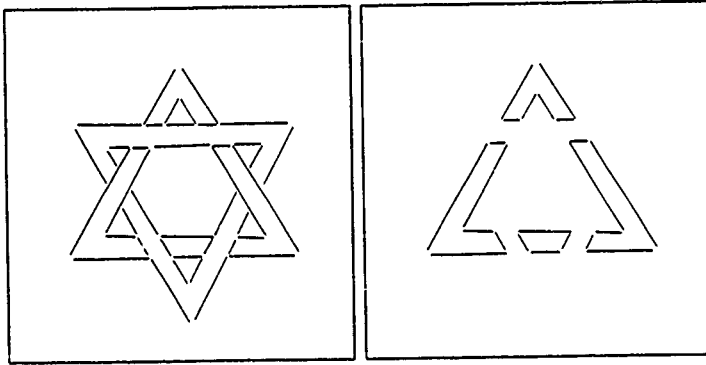


Figure 15: Image lines for intact star and fragments.

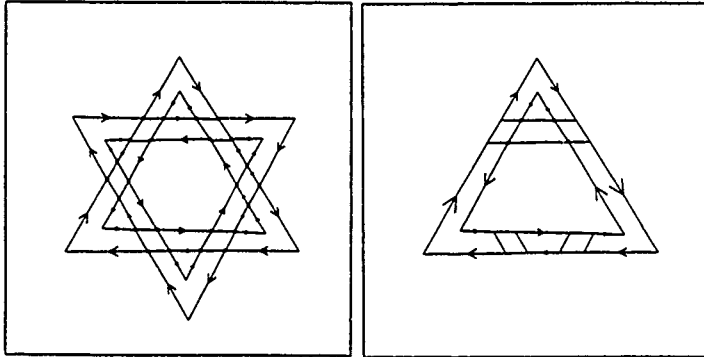


Figure 16: Contour graphs for intact star and fragments.

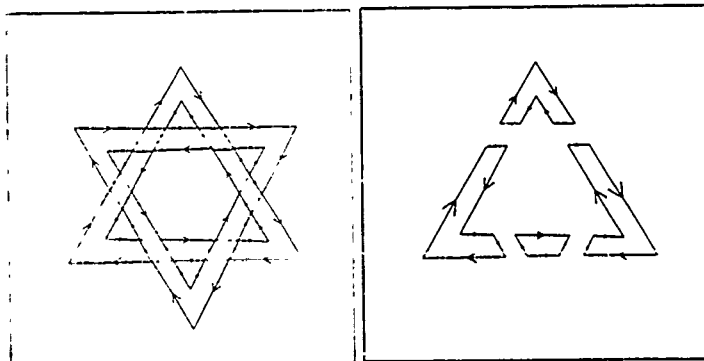


Figure 17: Boundary graphs for intact star and fragments.

designed to support intermediate level vision research. Among other things, the ISR provides: 1) Management of sets (and subsets) of image tokens; 2) Token pointers; 3) Fast spatial querying using grids; and 4) A file structure. Both the contour graph and the boundary graph are represented by ISR token sets containing endpoints, crossings, lines, corners and bars.

Table 1

Figure Name	Var.	Const.	B.B.	Pivots	Obj.
Triangle	447	465	1	209	296
Blocks	227	250	1	102	156
Intact Star	551	551	1	325	390
Fragments	187	183	3	134	131

The solutions to the integer linear programs were computed with a system called IMINOS [4]. IMINOS uses branch and bound search and the MINOS [16] linear programming package to solve integer linear programs. Table 1 contains a summary of the number of variables and constraints in the integer linear programs generated for the examples in this paper. The entries in the column labeled 'B.B.' are the number of branch and bound steps required to find the optimal feasible integer solution. Each branch and bound step requires an invocation of the Simplex algorithm. The column labeled 'Pivots' contains the *total* number of Simplex pivot steps required to solve the integer linear program. Although solving integer linear programs can be computationally prohibitive, the examples in this paper demonstrate that specific instances can be solved quickly and efficiently. None of the examples required more than 2 minutes of CPU time on a Sun 3/60.

4 Relationship to Blocks World

The current work differs from the majority of the blocks world work because its principal goal is grouping, rather than labeling. However, it exploits many of the same constraints and benefits from the notation.

In the late 1960's Adolpho Guzman [8] developed a program called SEE which grouped regions into single objects in blocks world scenes. The rules SEE used were heuristic and easily defeated. Nevertheless, grouping (including the completion of partially occluded objects) was the primary goal of the system.

Huffman [11] developed a label set which allowed legal blocks world scenes to be distinguished from impossible or "nonsense" scenes. One blocks world representation Huffman discusses is the labeled 'X-ray' picture, which is very similar to the boundary graph described in this paper. All visible and hidden lines in the 'X-ray' picture are labeled with a depth index. Subsequent blocks world work ignored the labeled 'X-ray' picture and concentrated on discovering consistent labels for the visible lines only. For example, Waltz filtering [26] is a systematic procedure for deriving a consistent labeling for the

visible lines in a blocks world scene. It assumes that a complete and accurate unlabeled scene graph already exists. Figural completion (i.e. insertion of hidden lines) was beyond its scope, yet there exist scenes which can not be uniquely labeled without it. Later, Rosenfeld, Hummel and Zucker [23] considered the problem of finding an *optimal* consistent scene labeling. This eventually led to the idea of a generalized *relaxation labeling* problem [12]. However, any program which inserts hidden lines in cluttered scenes (blocks world or otherwise) must keep track of the depth index of the lines as they are occluded and disoccluded. Otherwise, it is impossible to know which sections of line are visible and which are hidden, and hence, whether grouping is appropriate. This requires counting, and is more easily simulated in integer linear programming than in relaxation labeling.

5 Future Work

As mentioned earlier, the current system is only able to complete gaps in the straight sections of occluding contours, and is unable to complete figures with missing sides or corners. This is due to the fact that the system currently employs only one virtual line construction method when building the contour graph. The next logical step is to use curves to join contour fragments, rather than straight lines. This would permit a much broader class of completions. The curve of least energy suggested by Horn [10] is a good candidate. In addition, a grouping system for curvilinear contours [5] should provide the initial image tokens, replacing the straight line grouping system currently used. Together, these improvements will allow the system to group effectively under more general conditions.

Each of these extensions will doubtless increase the size of the optimization problems which must be solved. Certainly, massively parallel architectures will be of some help in offsetting the increased cost of deriving the contour graph from the image data and solving the larger integer linear programs which result. Additional constraints provided by unexploited figural cues (e.g. shadows and surface markings) can decrease computation time by reducing the size of the feasible region.

Part of the motivation for this work lay in correcting the shortcomings of current line grouping systems by exploiting knowledge of surfaces and occlusion in a principled way. It is hoped that in the near future, the system can be applied effectively to images of real scenes. As it stands, it is a working model of figural completion and illusory contour phenomena with implications for the study of human vision. I hope it is also a step toward reconciling perceptual organization and the goal of computing the $2\frac{1}{2}$ D Sketch.

Acknowledgements

Special thanks go to my advisor, Allen Hanson for his advice and encouragement. Thanks also to Harpreet Sawhney, Ugo Buy and Bruce Draper.

References

- [1] Barrow, H.G and Tenenbaum, J.M., Interpreting Line Drawings as Three Dimensional Surfaces, *Artificial Intelligence* 17, pp. 75-116, 1981.
- [2] Binford, T.O. Inferring Surfaces from Images, *Artificial Intelligence* 17, pp. 205-244, 1981.
- [3] Boldt, M., Weiss, R. and Riseman E.M., Token Based Extraction of Straight Lines, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 19, No. 6, pp. 1581-1594, 1989.
- [4] Burnett J., and U.Buy, Solving Integer Programming Systems Using the IMINOS Prototype, Technical Report, Department of Computer and Information Science, University of Massachusetts, Amherst, Mass., in preparation.
- [5] Dolan, J., Perceptual Grouping of Curved Lines, *Proceedings DARPA Image Understanding Workshop*, 1989.
- [6] Brolio, J., Draper, B., Beveridge, J.R. and Hanson A., ISR: A Database for Symbolic Processing in Computer Vision, *IEEE Computer* Vol. 22, No. 12, pp.22-30, 1989.
- [7] Burns, J.B., Hanson, A.R. and Riseman, E.M., Extracting Straight Lines, *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 8, No. 4, pp. 425-455, 1986.
- [8] Guzman, A., Computer Recognition of Three Dimensional Objects in a Visual Scene, , Ph.D. Dissertation, MIT, Cambridge, Mass., 1968.
- [9] Hoffman, D.D. and Richards, W., Parts of Recognition, *Cognition* 18, pp. 65-96, 1984.
- [10] Horn, B.K.P., The Curve of Least Energy, MIT AI Lab Memo No. 612, Artificial Intelligence Laboratory, MIT, Cambridge, Mass., 1981.
- [11] Huffman, D.A., Impossible Objects as Nonsense Sentences, *Machine Intelligence* 6, pp. 295-323, 1971.
- [12] Hummel, R.A., and Zucker, S.W., On the Foundations of Relaxation Labeling Processes, Internal Report TR-80-7, Dept. of Electrical Engineering, McGill University, Montreal, 1980.
- [13] Kanizsa, G.K., Subjective Contours, *Scientific American*, April 1976.
- [14] Koenderink J.J., The Shape of Smooth Objects and the Way Contours End, *Natural Computation*,

- W. Richards (ed.), MIT Press, Cambridge, Mass., 1988.
- [15] Marr, D., *Vision*, Freeman Press, San Francisco, Cal., 1982.
 - [16] Murtagh, B.A., and M.A.Saunders, MINOS 5.1 User's Guide, Technical Report SOL 83-20R, Department of Operations Research, Stanford University, Stanford, Cal., December 1983.
 - [17] Nevatia, R. and K.R. Babu, Linear Feature Extraction and Description, *Computer Graphics and Image Processing*, Vol. 13, pp. 157-269, 1980.
 - [18] Petry, S. and Meyer, G.E. (eds.), *The Perception of Illusory Contours*, Springer-Verlag, New York, 1987.
 - [19] Richards, W., Koenderink J.J. and Hoffman, D.D., Inferring 3D Shapes from 2D Silhouettes, *Natural Computation*, W. Richards (ed.), MIT Press, Cambridge, Mass., 1988.
 - [20] Rock, I., *The Logic of Perception*, MIT Press, Cambridge, Mass., 1983.
 - [21] Rock, I., *Perception*, Scientific American Books, New York, 1984.
 - [22] Rock, I., A Problem Solving Approach to Illusory Contours, *The Perception of Illusory Contours*, Petry and Meyer (eds.), Springer-Verlag, New York, 1987.
 - [23] Rosenfeld, A., Hummel, R.A., and Zucker, S.W., Scene Labeling by Relaxation Operations, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 6, No. 6, pp. 420-433, 1976.
 - [24] Stevens, K. and Brookes, A., Detecting Structure by Symbolic Constructions on Tokens, *Computer Vision, Graphics, and Image Processing* 37, pp. 238-260, 1987.
 - [25] Ullman, S., Filling-in the Gaps: The Shape of Subjective Contours and a Model for Their Generation, *Biological Cybernetics* 21, pp. 1-6, 1976.
 - [26] Waltz, D.L., Generating Semantic Descriptions of Scenes with Shadows, *The Psychology of Computer Vision*, P.H. Winston (ed.), McGraw-Hill Book Co., New York, 1972.
 - [27] Witkin, A. P. and Tenenbaum J.M., On the Role of Structure in Vision, *Human and Machine Vision*, Beck, Hope and Rosenfeld (eds.), Academic Press, 1983, pp. 481-543.

View variation of point set and line segment features

J. Brian Burns, Richard Weiss and Edward M. Riseman

Computer and Information Science Department
University of Massachusetts at Amherst *

Abstract

The recognition of 3D objects becomes much more difficult as the relative viewing position becomes less constrained. For an image feature to be effective in the discrimination of 3D objects, it is useful if the distribution of the feature values over permissible views tends to be narrow. This paper is a study of the variation of point-set and line-segment features with respect to view. It is first established that there is no general-case view-invariant defined for any number of points, given true perspective, weak perspective or orthographic projection models.

The remainder of the paper focuses on feature variation under weak perspective, a commonly used projection model in 3D recognition. Its special-case invariants are explained in terms of the invariance of linear dependence relations with respect to linear transformation. The variation with respect to view is then studied for an important set of 2D line segment features: the relative orientation, size and position of one line segment with respect to another. The analysis includes an important evaluation criterion for feature utility in terms of view-variation: the relationship between fraction of views (over a view sphere) and the range of values assumed by a feature over these views. This relationship is a function of both the feature and the particular configuration of 3D line segments; an analysis and series of graphs are presented for each of the features and for a few configurations of 3D line segments.

1 Introduction

One of the outstanding problems in visual object recognition is the fact that objects are usually three-dimensional structures, but they are typically sensed in the form of two-dimensional projections. While model-independent understanding of 3D structure [Marr82] is possible from motion, stereo, shading and texture, these cues may quite often be unavailable, unreliable or provide only a rough indication of the object structure. Hence it is important to develop systems capable of recognizing 3D objects by matching them to 2D image data.

The prediction-based methods developed in [Brooks81, Lowe85, Burns87, Korn87] constitute a promising approach to the recognition of 3D

objects in 2D images. In this approach, recognition is achieved by (1) predicting characteristics of the object projections from all views, (2) matching these predictions against the input 2D image and (3) verifying all promising matches by determining the 3D pose of the object given the data matched. In its most general form a *prediction* expresses the expected values for a set of features of the object's projection. A *feature* can be any measurement or function of the projection, and the expectations can be any valid statement about the feature distribution.

Another fundamental problem in recognition is ensuring that computational costs for recognition grow only slowly with respect to model base size. This can be achieved by organizing the predictions across objects into a discrimination structure [Grimson84, Ikeuchi87]. Such a structure, called a *prediction hierarchy*, has been developed in our research on recognition systems [Burns87]. Objects with shared predictions are progressively discriminated by specifying additional features whose value distributions are different for the different objects. After the discrimination structure is compiled, the recognition system uses it to recursively match predictions that are progressively more object-specific.

For an image feature to be useful in discrimination, its distribution of values with respect to each object should be narrow, and the distributions with respect to different objects should be well separated. Since the camera viewpoint may be only partially constrained, the usefulness of a feature over the object projection is a function of how it varies with view. Hence, the property of *view-variation*, the extent to which a feature varies over given ranges of view, is fundamental to the recognition process. Since features of a projection usually "blow-up" to extreme values at some (usually small) set of views, a feature is considered here to have *low view-variation* if the variation is small in extent over a large fraction of the views¹. Ideally, a feature should be *view-invariant*, that is, unaffected by change in view.

This paper presents a study of the variation of 2D point-set and line-segment features with respect to view. It is first established that there is no *general-case view-*

*This research was supported by the Defense Advanced Research Projects Agency under contract F30602-87-C-0140 and by Army ETL contract DACA76-89-C-0017.

¹The term *quasi-invariant* [Binford87] has been used to denote something similar. Unfortunately, it is used to denote other things as well, and is thus avoided here.

invariant defined for any number of points, given true perspective, weak perspective or orthographic projection models. That is, there does not exist a feature that is view-invariant for all point sets of a given size n , for any n . Instead, there are only *special-case* invariants, features that are only view-invariant for special configurations of 3D points. It is important to determine the existence of general-case invariants since their distribution of values for each object would always have zero variance; only added noise and small separation between the objects would hinder their usefulness. General-case invariants have been effectively used in domains where they do exist; for example, planar objects with 2D rigid transformation [Tucker88], and planar objects with 3D rigid transformation and weak perspective projection [Lamdan88a]. For the domain of 2D projections of 3D points, their existence does not seem to have been determined [Ahuja68, Duda73, Lowe85, Binford87, Aliomonos87, Lamdan88b, Weiss88]; in fact, in a chapter surveying projective geometry, Duda et al., explicitly mention the current lack of understanding on the subject.

The remainder of the paper then focuses on feature variation under weak perspective, a commonly used projection model in 3D recognition. Special-case invariants are surveyed for this projection model, and the variation with respect to view is studied for an important set of 2D line segment features: the relative orientation, size and position of one line segment with respect to another. The analysis includes an important evaluation criterion for feature utility in terms of view-variation: the relationship between fraction of views and the range of values assumed by a feature over these views. This relationship is a function of both the feature and the particular configuration of 3D line segments; an analysis and series of graphs are presented for each of the features and for a few configurations of 3D line segments.

2 General-case view invariants

In this section, it is established that there is no feature of the perspective projection of n points, for any n , that is both a general-case view-invariant and non-trivial in the senses defined below. This result is also extended to orthographic and weak perspective projection models.

2.1 Definitions

The perspective object-to-image transformation is $\pi_{R,\vec{T}}(\vec{P}) = (F/z) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} (R\vec{P} + \vec{T})$, for 3D point² \vec{P} , depth of point z , 3D rotation R , 3D translation \vec{T} and focal length F . Each (R, \vec{T}) represents a distinct view.

The projection of the 3D point set S will be compactly represented as $\pi_{R,\vec{T}}(S)$, and the value for a given feature f , view (R, \vec{T}) and 3D point set S is $f(\pi_{R,\vec{T}}(S))$. Since we will be dealing with multiple point sets and these sets

²Henceforth, lower case signifies image points and upper case, object points.

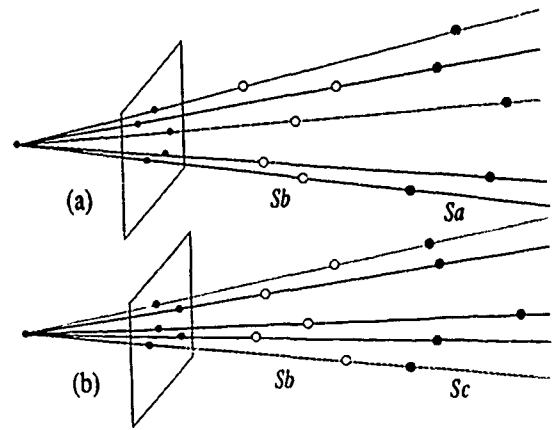


Figure 1: Perspective and projective correspondence. Point sets S_a and S_b , and S_b and S_c , are in perspective correspondence; point sets S_a and S_c are in projective correspondence through S_b .

are ordered, we will refer to the j th point of the i th set as $P_{i,j}$.

Def: A feature f is a *general-case* view-invariant for the class of all 3D point sets C^n if $\forall S \in C^n. \forall (R_1, \vec{T}_1), (R_2, \vec{T}_2), f(\pi_{R_1, \vec{T}_1}(S)) = f(\pi_{R_2, \vec{T}_2}(S))$ ³.

There is always a trivial f that can satisfy the above property: any constant feature, where a *constant* feature has the same value for all 3D point sets and views. Thus, the following is another important property for a feature.

Def: A feature f is *non-trivial* if there exist two different point sets S_1 and S_2 , and a pair of views (R_1, \vec{T}_1) and (R_2, \vec{T}_2) , such that $f(\pi_{R_1, \vec{T}_1}(S_1)) \neq f(\pi_{R_2, \vec{T}_2}(S_2))$.

The following are two concepts from projective geometry [Duda73] that must be adapted to our problem domain of 3D point sets and 2D projections.

Def: Perspective correspondence. Two point sets are in perspective correspondence if there exists a pencil of rays that pass through every point in each set, and every ray in this pencil passes through the same number of points from each set⁴ (Figure 1a). Clearly, two such 3D point sets project to the same image if the camera focal point is the ray intersection point and the rays are sectioned by the image plane; in other words, $\pi_{R_a, \vec{T}_a}(S_a) = \pi_{R_b, \vec{T}_b}(S_b)$, for some $(R_a, \vec{T}_a, R_b, \vec{T}_b)$.

Def: Projective correspondence. Two point sets are in projective correspondence if they can be connected by a chain of perspective correspondences. For example, S_a and S_c are in projective correspondence through perspective correspondences with S_b . This is depicted in Figure 1 by keeping the focal, or ray pencil point fixed and rotating the 3D point set S_b to align it with rays passing through S_a and S_c respectively.

³We are assuming f is defined for all but a measure zero set of views and point-sets (*degenerate views* for f)

⁴Additionally, all rays must be constrained to the same half-space to create an image by sectioning the pencil with a plane

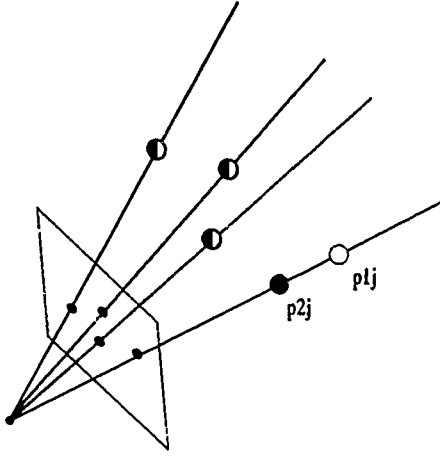


Figure 2: 3D point sets S_1 (white) and S_2 (black) are identical up to $\vec{P}_{1,j}$ and $\vec{P}_{2,j}$. To show that such sets are in perspective correspondence select a focal point in the line through $\vec{P}_{1,j}$ and $\vec{P}_{2,j}$; in this way each pair of corresponding points, one from each set, share a ray.

2.2 Theorem and proof

Lemma 1 A general-case view-invariant has the same value for point sets in projective correspondence⁵.

Proof. For point sets in perspective correspondence S_1 and S_2 , this is immediate. Being a general-case invariant, f has a single value f_1 for all projections of S_1 , and similarly for S_2 (f_2). Since $\pi_{R_1, \vec{T}_1}(S) = \pi_{R_2, \vec{T}_2}(S)$, for some pair of views $(R_1, \vec{T}_1, R_2, \vec{T}_2)$, we have $f_1 = f(\pi_{R_1, \vec{T}_1}(S_1)) = f(\pi_{R_2, \vec{T}_2}(S_2)) = f_2$.

By definition, for point sets in projective correspondence S_1 and S_n , there exists a sequence of point sets S_2, \dots, S_{n-1} such that every pair (S_i, S_{i+1}) is in perspective correspondence, for $1 \leq i < n$. By transitivity of equality, the feature values associated with S_1 and S_n must be equal. \square

Lemma 2 Two 3D point sets of the same size are in perspective correspondence if they are identical for all but one point⁶.

Proof. Consider two such sets S_1 and S_2 , and say that they differ in the position of the j th point; i.e., $\vec{P}_{1,j} \neq \vec{P}_{2,j}$ (Figure 2). In order to show perspective correspondence between the sets, it is sufficient to show that there exists a pencil of rays that contains both sets. This can be done by placing the focal point (point of intersection for the rays) in the line that passes through $\vec{P}_{1,j}$ and $\vec{P}_{2,j}$. Clearly these two points will then have a ray in common and the other point pairs will share rays by virtue of the fact that they occupy the same positions in space. \square

Lemma 3 All 3D point sets of size n are in projective correspondence.

Proof. Consider two sets of size n , S_1 and S_n . For these two sets, construct a sequence of sets (S_2, \dots, S_{n-1}) such that, for $1 \leq j < n$, adjacent pairs S_j and S_{j+1} are identical, except for their j th points. This sequence can be constructed by defining each S_j to be the union of the subset $\{\vec{P}_{1,i} | 1 \leq i < j\}$ of S_1 and the subset $\{\vec{P}_{n,i} | j \leq i \leq n\}$ of S_n .

From Lemma 2, every adjacent pair of point sets in this sequence is in projective correspondence, and, by transitivity, the original point sets S_1 and S_n are also. \square

Theorem 1: there is no feature of n projected points, for any n , that is both a general-case view-invariant and non-trivial in the senses defined above.

Proof. Consider a general-case view-invariant f . For f to be non-trivial, there must exist at least one pair of point sets S_1 and S_2 , such that $f(\pi_{R_1, \vec{T}_1}(S_1)) \neq f(\pi_{R_2, \vec{T}_2}(S_2))$ for some $(R_1, \vec{T}_1, R_2, \vec{T}_2)$.

Since all 3D point sets of the same size are in projective correspondence (Lemma 3) and every pair in projective correspondence must have the same value for any general-case view-invariant (Lemma 1), the necessary condition for a non-trivial feature cannot be established for any such general-case view-invariant. \square

2.3 Extension to other projection models

Theorem 1 also holds for two other commonly used projection models: orthographic and weak perspective. Orthographic projection is the same as perspective, except that the projection rays must always intersect at a point at infinity. Only Lemma 2 considers the projection geometry, and since its construction step allows the intersection point to lie anywhere in a given line, the point can always be placed at infinity. Weak perspective is identical to orthographic except that the projection is also scaled by the average depth of the 3D point set. Since the added scaling simply represents another degree of freedom in the transformation, the lack of general-case invariants under orthography must imply the same for weak perspective.

3 Weak perspective projection

The *weak perspective* projection model is used in our research for the purposes of predicting image properties. It is an approximation to perspective projection that is applied extensively in object recognition research as it simplifies the analysis and computation for 3D object recognition with reasonable results when the camera is far enough from the object relative to the depth variation of the object [Brooks81, Thompson87, Huttenlocher87, Lamdan88a]. In weak perspective, all of the points on a 3D object are treated as being at the same distance from the camera: $(x_i, y_i) = F/z_0(X_i, Y_i)$, for average distance z_0 . Given this, the object-to-image transformation becomes much simpler: there is a single scale factor F/z_0 , instead of a different variable z_i for each point, each with a non-linear effect. Another reason for working with a

⁵Established through views not degenerate for f

⁶Perspective correspondence cannot always be established through views non-degenerate for a given feature f . However, a projective correspondence can, by suitable intermediate point sets, and is sufficient to establish Theorem 1 [Burns90].

weak perspective model is that there are some useful special-case view-invariants (Section 4) that are approximately invariant for true perspective at an appropriately large camera distance. For example, parallelism can be effectively used as a line grouping criterion in grouping-based recognition [Lowe85]

It is important to note that prediction and match errors generated by assuming weak perspective have yet to be suitably analyzed. Some understanding can be gained by examining errors in the image point position. Consider a point in space (X, Y, Z) on an object with average camera distance z_0 . For perspective projected x and weak perspective approximation \hat{x} , we have

$$(x - \hat{x})/x = FX(1/z - 1/z_0)/(XF/z) = (z_0 - z)/z_0$$

(similarly for y). Thus the proportional error in image position is equal to the relative depth $(z_0 - z)$ over the absolute average depth z_0 . This depth ratio is the one that Thompson et al., recommend to be under one tenth; at this range, the error in image position is at ten percent. This may be quite acceptable for recognition systems that use weak perspective to predict rough ranges in the orientation, size and position of image features. This is especially true if multiple features are used in discrimination and the objects are reasonably different. However, it is important to remember that a weak perspective approximation can contribute non-trivial error.

4 Special-case view-invariants under weak perspective

The weak perspective object-to-image transformation is a singular affine transformation that can be represented by a 2×3 rank 2 matrix and 2D translation vector [Lamdan88b]. By considering only point position differences, the translation component can be subtracted out and the vectors representing the point position differences can be related to their projections through a linear transformation. This is important since the property of linear dependence for a set of vectors, the coefficients expressing the linear dependence, and the subspace dimension associated with the dependence are all invariant to linear transformation.

All of the weak perspective invariants discussed in the recognition literature [Brooks81, Lowe85, Lamdan88b] follow from this observation, and other, related special-case invariants can be deduced that may also be useful for recognition algorithms. For each type of special-case invariant, the required 3D point-set conditions can be specified as the linear dependence of some set of their position differences, and the invariant feature itself can be defined as a coefficient or subspace dimension associated with this linear dependence.

The different special-case conditions can be distinguished by two properties: the dimension of the subspace containing the set of 3D difference vectors, and the pattern of the point differences. The subspace dimensions given 3D point sets are two, one, and zero, the point difference patterns specify which points are being subtracted into which other points. Two types of difference

Differences from single reference point: $\{(\vec{p}_i - \vec{p}_1) i \geq 2\}$	Paired-off point differences $\{(\vec{p}_{2i} - \vec{p}_{2i-1}) i \geq 1\}$
Special-case constraint: Four 3D points such that $\sum_{i=2}^4 c_i(\vec{p}_i - \vec{p}_1) = \vec{0}$ for non-zero c_i (4 planar \vec{p}) View invariants: (1) $(c_3/c_2, c_4/c_2)$. Example: affine coordinates [Lamdan88a] (2) Subspace D (=2), not significant for 2D images.	Special-case constraint: Six 3D points such that $\sum_{i=1}^3 c_i(\vec{p}_{2i} - \vec{p}_{2i-1}) = \vec{0}$ for non-zero c_i . (The \vec{p} need not be coplanar.) View invariants: (3) $(c_2/c_1, c_3/c_1)$. No known example of use (4) Subspace D (=2), not significant for 2D images.
Special-case constraint: Three 3D points such that $\sum_{i=2}^3 c_i(\vec{p}_i - \vec{p}_1) = \vec{0}$ for non-zero c_i . (3 collinear \vec{p}) View invariants: (5) c_3/c_2 . Example: approach ratio in [Brooks81]. (6) Subspace D (=1). Example: collinearity.	Special-case constraint: Four 3D points such that $\sum_{i=1}^2 c_i(\vec{p}_{2i} - \vec{p}_{2i-1}) = \vec{0}$ for non-zero c_i . (Endpoints of two parallel line segments.) View invariants: (7) c_2/c_1 . Example: parallel distance ratio (8) Subspace D (=1), Example: parallelism, used in grouping [Lowe85].
Special-case constraint: Two 3D points such that $c_2(\vec{p}_2 - \vec{p}_1) = \vec{0}$ for non-zero c_2 . (2 coincident \vec{p}). View invariants: (9) c_2 . Uninteresting (can be anything but zero). (10) Subspace D (=0). Example: coincidence, used in grouping [Lowe85].	Same case as left panel.

Table 1: Classification of special-case view-invariants under weak perspective. All invariants discussed in the literature are functions of point position differences, and they can be distinguished by two properties: the dimension of the subspace containing the set of 3D difference vectors (row), and the pattern of the point differences (column).

patterns can be found in the invariants surveyed: differences from a single reference point \vec{p}_1 selected from the set, or $\{(\vec{p}_i - \vec{p}_1) | i \geq 2\}$, and differences between points that have been paired off, $\{(\vec{p}_{2i} - \vec{p}_{2i-1}) | i \geq 1\}$.

Table 1 shows the classification of weak-perspective invariants based on these two properties of their special-case conditions. It is interesting to note that special-case invariants can be deduced from this framework that have not been found in the recognition literature, for example, invariant (3) classified under (*dimension two, paired-off point differences*).

There are also special-case invariants under perspective projection [Duda73, Lamdan88b]; however, they seem to only correspond to a subset of the above features and tend to require more points. The cross-ratio is analogous to the distance ratio (5), but requires four points instead of three, and projective coordinates correspond to the affine coordinates (1), but require five points instead of four [Duda73]. There do not seem to be perspective invariants analogous to the ones in the second column (3, 7, 8), requiring linear dependence of paired-off point differences.

5 View variation of relative orientation, size and position

Though view-invariance is restricted to certain special 3D line segment configurations, the property of low view-variation, as defined in Section 1, is much more common. It is also of practical importance, since a reasonably small variation in feature value for each object, relative to the value variation *across* the objects, can be quite effective for 3D object discrimination in the sense discussed in Section 1. It is difficult to be precise about what we mean by *low* view-variation as it is context dependent, being meaningful in terms of the feature value distributions over the particular set of objects being discriminated. However, we can be precise about the extent of variation of a feature as a function of particular 3D line segment configurations and portions of view space.

This section presents an analysis of view-variation for four important features of a projected line segment pair. They are the *relative* orientation, size and position of one segment with respect to the other. These features are defined and motivated in the next subsection. For each feature, the following results are presented:

- *The feature value as a function of view and 3D line segment configuration.* The value is expressed as a function only of the view parameters and parameters of the 3D segment configuration that affect it. The expression is simplified as much as possible by using an appropriate object coordinate frame and coordinate transformation representation.
- *Qualitative analysis.* This includes a description of the 3D line segments and ranges of views for which the feature variation is most and least constrained, and what the feature values are at these points.
- *Quantitative analysis.* Graphs are presented and discussed showing the feature variation as a function of view and 3D segment configuration⁷. In addition, the relationship between extent of view-variation, the 3D segments and the range of views is explicitly presented. Feature interval size is plotted against view region size for various 3D segments.

5.1 The features

The view-variation of four features are studied in this section: the orientation, size and position (two components) of one projected line segment with respect to another (Figure 3). These *relative* features can be pictured in the following way (Figure 3b). First, a coordinate frame is defined in terms of the first projected segment: the origin is set at its first endpoint, one axis (u) is aligned with the segment, the other is orthogonal to it (v) and the scale is such that the first segment is of unit length. The orientation, position and size of the second segment is then measured with respect to this coordinate frame. Specifically, given the projected segments (\vec{p}_1, \vec{p}_2) and (\vec{p}_3, \vec{p}_4) , the four features are defined in the following manner:

⁷Due to space, this is only presented here for α , see [Burns90] for a complete set

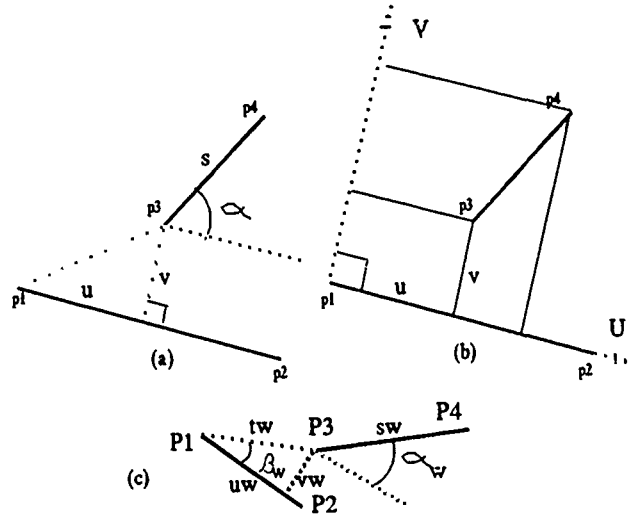


Figure 3: The relative features of line segments: (a) image segments (bold) and the projected relative orientation α , size s and position (u, v) , (b) coordinate frame (U, V) aligned with first segment and used to measure second segment, (c) corresponding 3D segments and features.

- relative orientation α . This is measured counter-clockwise from $(\vec{p}_2 - \vec{p}_1)$ to $(\vec{p}_4 - \vec{p}_3)$ and has a magnitude of $\arccos((\vec{p}_2 - \vec{p}_1)(\vec{p}_4 - \vec{p}_3) / |\vec{p}_2 - \vec{p}_1| |\vec{p}_4 - \vec{p}_3|)$,
- relative size s is the length ratio, $|\vec{p}_4 - \vec{p}_3| / |\vec{p}_2 - \vec{p}_1|$,
- relative position (u, v) is the position of endpoint \vec{p}_3 relative to the segment (\vec{p}_1, \vec{p}_2) . It is the displacement of \vec{p}_3 from \vec{p}_1 measured along and normal to $(\vec{p}_2 - \vec{p}_1)$ and divided by its length, or $u = (\vec{p}_3 - \vec{p}_1)(\vec{p}_2 - \vec{p}_1) / |(\vec{p}_2 - \vec{p}_1)|^2$ and $v = (\vec{p}_3 - \vec{p}_1)(\vec{p}_2 - \vec{p}_1)^\perp / |(\vec{p}_2 - \vec{p}_1)|^2$, where $(\vec{p}_2 - \vec{p}_1)^\perp$ is $(\vec{p}_2 - \vec{p}_1)$ rotated by 90 degrees in the image plane.

For each of the relative 2D features of the projected segments there is a corresponding parameter of the 3D segment configuration being projected; these 3D world parameters are indicated by: α_w, s_w, u_w, v_w (Figure 3c). In addition, to analyze (u, v) , it is useful to keep in mind β_w , the angle between $(\vec{P}_2 - \vec{P}_1)$ and $(\vec{P}_3 - \vec{P}_1)$, and t_w , the distance between \vec{P}_3 and \vec{P}_1 scaled by the length of $(\vec{P}_2 - \vec{P}_1)$.

5.2 View representation

Each feature value is a function of view and the 3D segments; however only certain parameters of the view and 3D segment configuration affect this value. In this section, we define and justify a simplified expression of the weak perspective object-to-image transformation that is only in terms of the view parameters relevant to the discussed features.

All of the features considered here are invariant to rotation about the optical z axis and all three degrees of freedom in translation. Therefore, the only aspect of the view transformation that affects these features is the orientation of the optical z axis relative to the object coordinate system, say the object Z axis. The

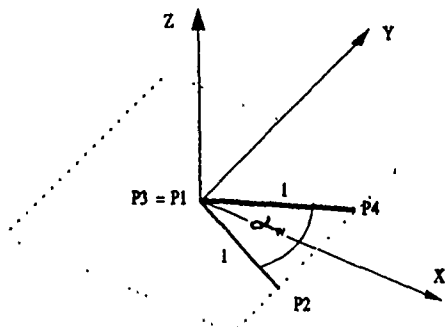


Fig. 4 View sphere coordinates (ϕ, θ) expressing the orientation of optical z axis relative to object Z (north pole). The angle ϕ is the azimuth (angle about the pole) and θ is the angle from the pole (90 - elevation).

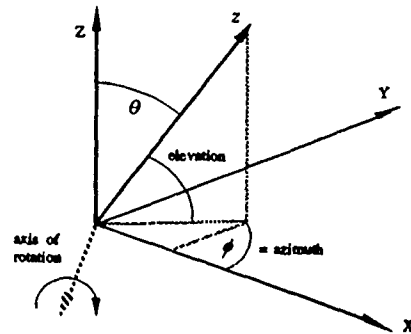


Fig. 5 3D line segment configuration with interior angle α_w . The view variation of α can be studied by analyzing the projections of this pair.

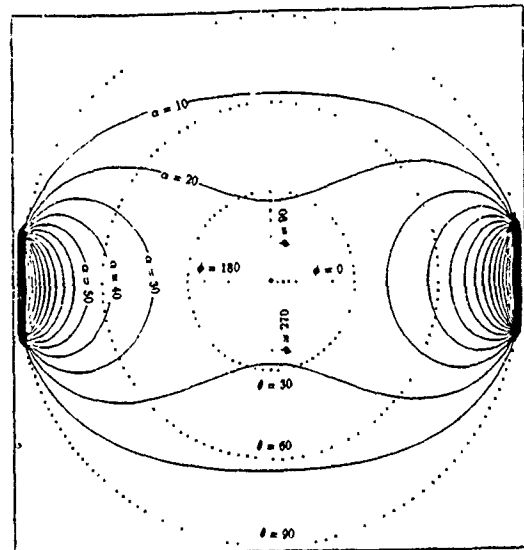
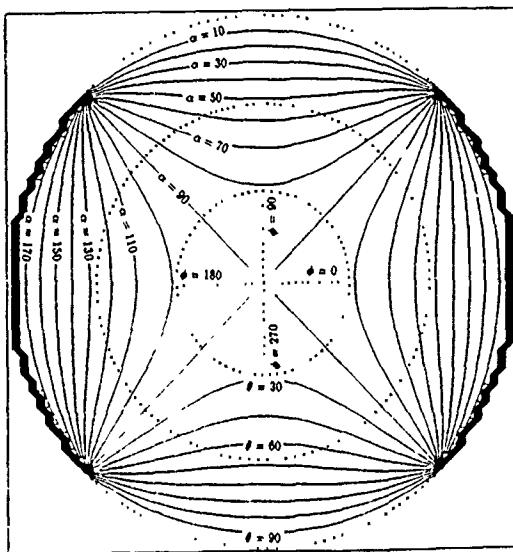


Fig. 6a,b Polar plots of α as a function of view (θ, ϕ), for (a) 3D angle $\alpha_w = 90$ and (b) 22.5 degrees. The angle ϕ is represented by the clockwise angle about the center of the plot, with $\phi = 0$ at the three o'clock position, and θ is represented by the radius out from the center. The contour lines are constant values of α (slices of the surface at various α).

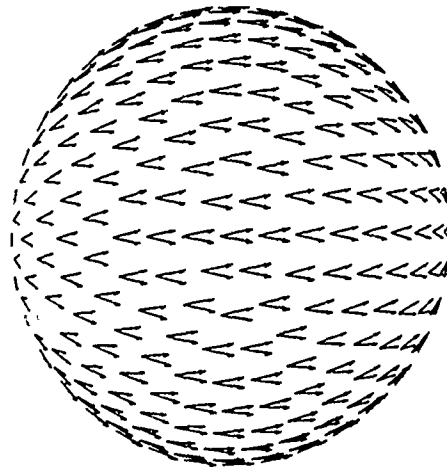
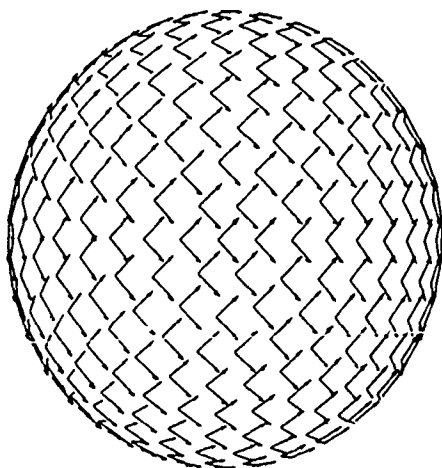


Fig. 6c,d Projections of the 3D line segments in Figure 5 from different views and for (a) 3D angle $\alpha_w = 90$ and (b) 22.5 degrees. Each projection is plotted on the sphere at roughly the view position (θ, ϕ) from which the segments would be seen this way. The sphere is orthographically projected, with the object Z is pointing out of the page and the X axis pointing to the left.

relative orientation of z can be expressed in terms of two parameters (ϕ, θ) , which represent the positions of on a unit view sphere about the object with the north pole at Z (See [Horn86] and Figure 4). The angle ϕ is the azimuth (angle about the pole) and θ is the angle from the pole (90° - elevation). The optical z axis in object coordinates is then $(\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta)$. Using this representation, the essential rotation from object to image can be expressed in terms of the two view sphere coordinates: the magnitude of the rotation from Z to z is simply θ , and the axis of the rotation $z \times Z / |z \times Z| = (\sin \phi, -\cos \phi, 0)$. We will refer to this rotation as $R_{\phi, \theta}$. The object-to-image transformation under consideration can now be expressed as $\vec{p}_i = \pi R_{\phi, \theta} \vec{P}_i$, where π is orthographic projection.

5.3 Relative orientation, α

In this section we first express the relative orientation of two projected segments as a function of (ϕ, θ) and the relevant parameters of the 3D segments. The magnitude of α is simply the angle between the two projected segments or $\arccos((\vec{p}_2 - \vec{p}_1)(\vec{p}_4 - \vec{p}_3) / |\vec{p}_2 - \vec{p}_1| |\vec{p}_4 - \vec{p}_3|)$ for projected endpoints \vec{p}_i .

The feature α is strictly a function of the projected differences $(\vec{p}_4 - \vec{p}_3)$ and $(\vec{p}_2 - \vec{p}_1)$, and, for weak perspective projection, these differences are unaffected by translation of the 3D segments relative to the object center and to each other. Also, the angle between the projected segments is unaffected by the lengths of the 3D segments. Thus, the only two factors affecting α are the orientation of the segments with respect to the camera (ϕ, θ) and the angle between them α_w . Thus, without loss of generality, we can analyze the variation of α by studying the projections of the 3D segment pair (see Figure 5):

$$\begin{aligned} P_1 &= P_3 = (0, 0, 0), \\ P_2 &= (\cos(\alpha_w/2), -\sin(\alpha_w/2), 0), \\ P_4 &= (\cos(\alpha_w/2), \sin(\alpha_w/2), 0), \end{aligned}$$

where α_w is the angle between the 3D segments. The feature α can then be expressed as a function of the view parameters (ϕ, θ) defined in Section 5.2 and the 3D angle α_w :

$$\begin{aligned} \alpha &= \arccos(\vec{p}_2 \vec{p}_4 / |\vec{p}_2| |\vec{p}_4|), \\ \vec{p}_2 &= \pi R_{\phi, \theta} (\cos(\alpha_w/2), -\sin(\alpha_w/2), 0)^T \\ \vec{p}_4 &= \pi R_{\phi, \theta} (\cos(\alpha_w/2), \sin(\alpha_w/2), 0)^T \end{aligned}$$

Figures 6(a,b) show polar plots of α as a function of view (ϕ, θ) , given the above 3D line segments with (a) $\alpha_w = 90^\circ$ and (b) 22.5° . The plots are over a hemisphere of views, where ϕ ranges from 0 to 360° and θ , from 0 to 90° . The angle ϕ is represented by the clockwise angle about the center of the plot, with $\phi = 0$ at the three o'clock position, and θ is represented by the radius out from the center. The contour lines are constant values of α (slices of the surface at various α). A sense of where the variation is greatest, and by how much, can be gained by observing the contour line density: the denser the lines, the greater the variation in feature α with respect to change in view (ϕ, θ) . For

example, the variation is greatest when the view direction approaches the orientation of the 3D line segments (bold lines) and slowest when oriented normal to the plane containing the line segments (xy plane).

Figures 6(c,d) show a less quantitative but more intuitive picture of the view variation for the same 3D angles. In this figure, the actual line segment projections are shown for various views about the sphere parametrized by (θ, ϕ) . Each projection is plotted on the sphere at the view position from which it would be seen this way, and α is the angle between the projected lines at that view position. The sphere is orthographically projected, with the object Z pointing out of the page and the X axis pointing to the left. Additional plots, for different values of α_w , can be found in [Burns90].

From the equations and plots, the following observations can be made:

- When α_w is zero (i.e., the 3D segments are parallel) then α is constant (zero) for all (θ, ϕ) .
- As α_w approaches zero, the variation as a function of (θ, ϕ) is slow over most of the views.
- When $\alpha_w > 0$, there always exists some view where $\alpha = 180^\circ$ and $\alpha = 0^\circ$. In other words, α covers the full range of possible values. Angle α approaches 0 when (θ, ϕ) approach $(90, 90)$ and $(90, -90)$, and α approaches 180 when (θ, ϕ) approach $(90, 0)$ and $(90, 180)$.
- α is slowest in variation when the view is most parallel to the Z axis and normal to the plane containing the segments ($\theta = 0$). At this point $\alpha = \alpha_w$. The variation in α is still fairly slow for views near the object Z axis, especially if the 3D angle α_w is reasonably small.

A more quantitative picture can be gained by studying the distribution of α as (ϕ, θ) varies, for the different 3D angles α_w . Using an approximately regular sampling of the sphere of viewpoints parametrized by (ϕ, θ) , histograms of the number of views that fall within regular intervals of α were made for various 3D angles α_w (Figure 7). For example, the wider, symmetric graph (light gray) represents the distribution of α over all views for 3D line segments with angle $\alpha_w = 90^\circ$. The projected feature α ranges from 0 to 180° , and for the histograms, this range was divided into fifty regular intervals. The view-sphere sampling was done so that the views are approximately one degree apart⁸.

As the histograms show, the distributions of the projected features are strongly concentrated about the value of the true 3D angles, α_w , and this concentration becomes more pronounced as α_w approaches zero (similarly for 180°). An even more revealing picture can be gained by counting the number of views for which α falls within some interval about the 3D angle α_w ; i.e.,

⁸The actual sampling produced 20447 samples for a hemisphere, with an average angle between adjacent ones of 0.9999° and standard deviation of 0.001° . (For discussion on the sampling technique, see the appendix of [Burns90])

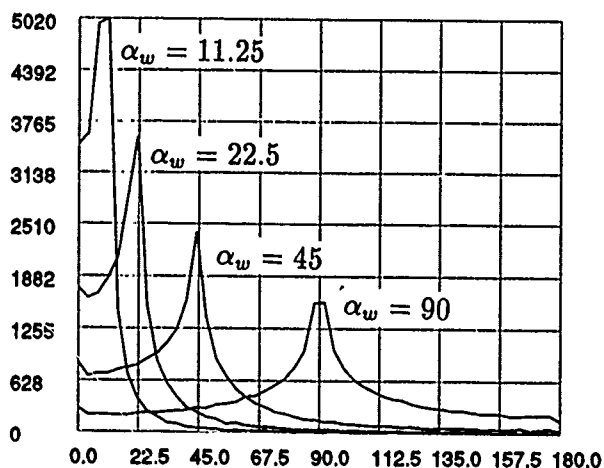


Fig. 7 Histograms of the number of views that fall within regular intervals of α for a hemisphere of views and for objects with various true 3D angles, α_w . (The range of α is divided into fifty intervals.)

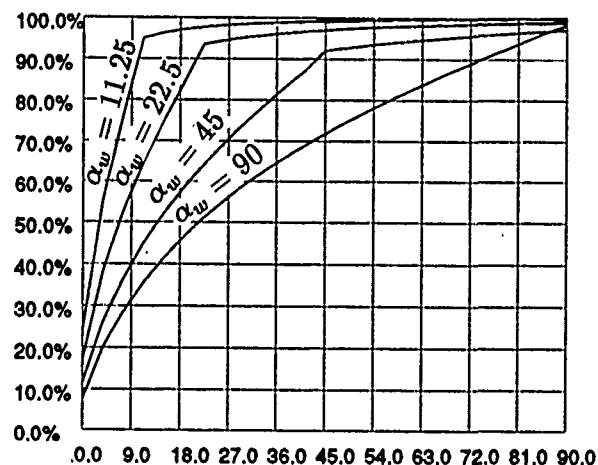


Fig. 8 The percentage of the view sphere that falls within the interval $[\alpha_w - \Delta\alpha, \alpha_w + \Delta\alpha]$ for various $\Delta\alpha$ ranging from 0 to 90 (degrees) and α_w of 90, 45, 22.5 and 11.25 (degrees).

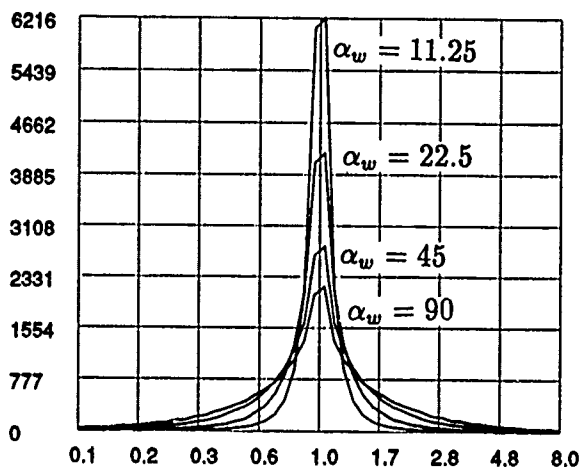


Fig. 9 Histograms of the number of views that fall within regular intervals of $\log_2 s$ for a hemisphere of views and for objects with various true 3D angles, α_w (and true $s_w = 1$). (The range of s is divided into fifty intervals of \log_2 spacing and the plot uses a \log_2 scale for s .)

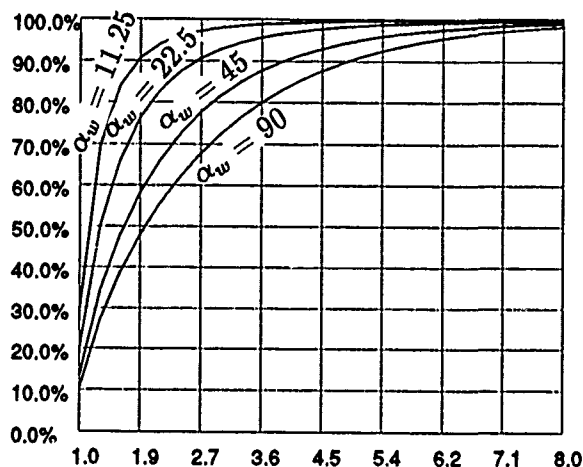


Fig. 10 The percentage of the view sphere that falls within the interval $[1/\Delta s, \Delta s]$ for various Δs ranging from 1 to 8 and α_w of 90, 45, 22.5 and 11.25 (degrees).

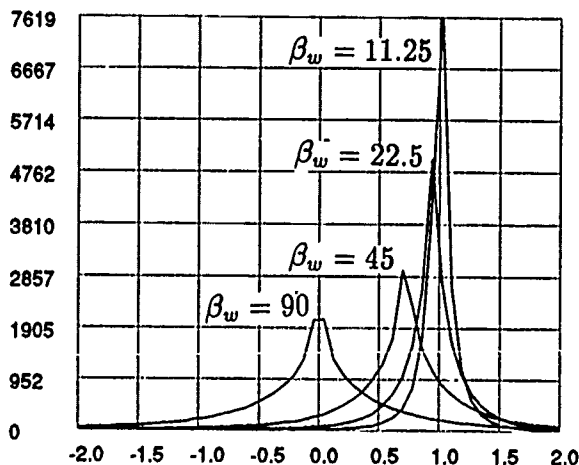


Fig. 11 Histograms of the number of views that fall within regular intervals of u for a hemisphere of views and for objects with various true 3D angles, β_w . (The range of u is divided into fifty intervals.)

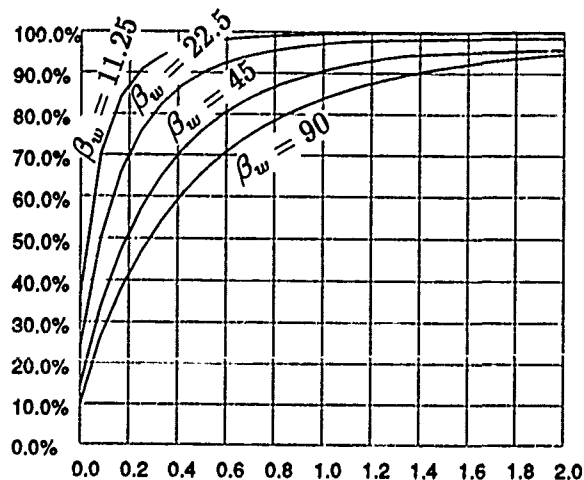


Fig. 12 The percentage of the view sphere that falls within the interval $[u_w + \Delta u, u_w - \Delta u]$ for various Δu ranging from 0 to 2 and β_w of 90, 45, 22.5 and 11.25 (degrees).

$[\alpha_w - \Delta\alpha, \alpha_w + \Delta\alpha]$. Figure 8 shows the percentage of the view-sphere that falls within this interval for various $\Delta\alpha$ ranging from 0 to 90 degrees and α_w of 90, 45, 22.5 and 11.25 degrees. The amount of the view sphere that falls within an interval clearly gets larger as α_w decreases. For the values of α_w studied, a smallish interval of $[\alpha_w - 15, \alpha_w + 15]$ covers approximately 42, 53, 75 and 96 percent of the sphere, respectively. To guarantee that most of the sphere (say 80 percent of it) falls within the interval, $\Delta\alpha$ has to be approximately 55, 34, 16 and 8 degrees for each of the α_w , respectively.

5.4 Relative size, s .

Relative size, s , is the ratio of the two projected segment lengths or, $s = |\vec{p}_4 - \vec{p}_3|/|\vec{p}_2 - \vec{p}_1|$. This feature is affected by the same view and 3D line segment structure parameters as α , for the same reasons, except that s is also clearly affected by the ratio of the lengths of the 3D segments, s_w . Thus, 2D feature s is strictly a function of view (ϕ, θ) , the angle between the 3D line segments α_w and s_w . Without loss of generality, this function can be represented by considering the same 3D line segment configuration as for α , except that \vec{P}_4 is scaled by s_w . By considering the projections of this configuration, we get the following expression for s :

$$\begin{aligned} s &= |\vec{p}_4|/|\vec{p}_2|, \\ \vec{p}_2 &= \pi R_{\phi, \theta}(\cos(\alpha_w/2), -\sin(\alpha_w/2), 0)^T \\ \vec{p}_4 &= s_w \pi R_{\phi, \theta}(\cos(\alpha_w/2), \sin(\alpha_w/2), 0)^T \end{aligned}$$

From the above equations, s is simply a linear function of s_w . The variation of s as the view varies, for different α_w , can be appreciated by observing the ratio of lengths of the projected line segments in Figure 6(c,d). Contour plots of s as a function of view (ϕ, θ) , for different α_w , can be found in [Burns90]. From the equations and plots, it is clear that:

- When α_w is zero (i.e., the 3D segments are parallel) then s is constant (s_w) for all (θ, ϕ) .
- As α_w approaches zero, the variation as a function of (θ, ϕ) is slow over most of the views.
- When $\alpha_w > 0$, there always exists some view where $s = \infty$ and $s = 0$. In other words, as in the case of α , s covers the full range of possible values. The value s approaches 0 when (θ, ϕ) approach $(90, \alpha_w/2)$ and $(90, 180 + \alpha_w/2)$. This is when the view aligns with the second segment. It approaches ∞ when (θ, ϕ) approach $(90, -\alpha_w/2)$ and $(90, 180 - \alpha_w/2)$ - when the view lines up with the first segment.
- As in the case of α , s is slowest in variation when the view is most normal to the plane containing the segments ($\theta = 0$). At this point $s = s_w$. Also, variation in s is still fairly slow for views near the Z axis, especially if the 3D angle α_w is reasonably small.

Figure 9 shows the histograms of the number of views that fall within regular intervals of s for various 3D angles of α_w , with s_w set to one and the same view sampling scheme as in Section 5.3. The projected size ratio

s actually ranges from zero to infinity - but for these examples, the histograms range from zero to eight and contain almost all of the 20,477 samples. For reasons of symmetry, the s axis is plotted using a \log_2 scale⁹.

As the histograms show, the distributions of the projected features are strongly concentrated about the value of the true 3D length ratio (in this case $s_w = 1$), and this concentration becomes more pronounced as α_w approaches zero (similarly for 180 degrees). A more revealing picture can be gained by counting the number of views from which s falls within some interval about the true feature $s_w = 1$. Again, for reasons of symmetry, the interval is taken to be $[1/\Delta s, \Delta s]$.

Figure 10 shows the percentage of the view-sphere that falls within this interval for various Δs ranging from 1 to 8 and α_w of 90, 45, 22.5 and 11.25 degrees. The amount of the view sphere that falls within an interval clearly gets larger as α_w decreases. For the values of α_w studied, the smallish interval of $[1/2, 2]$ covers approximately 53, 64, 80 and 93 percent of the sphere respectively. To guarantee that most of the sphere (say 80 percent of it) falls within the interval, Δs has to be approximately 3.6, 2.8, 2 and 1.5 for decreasing values of α_w respectively.

5.5 Relative position, u

Relative position comes in two components (u, v) measured in orthogonal directions in the image plane (Section 5.1). Due to space considerations, we will present an analysis of the view variation of u only, and refer the reader to [Burns90] for a discussion of v . The feature $u = (\vec{p}_3 - \vec{p}_1)(\vec{p}_2 - \vec{p}_1)/|(\vec{p}_2 - \vec{p}_1)|^2$.

The 2D relative position features are functions of the vectors $(\vec{p}_2 - \vec{p}_1)$ and $(\vec{p}_3 - \vec{p}_1)$ instead of $(\vec{p}_2 - \vec{p}_1)$ and $(\vec{p}_4 - \vec{p}_3)$. This means that the situation is exactly the same as for the 2D feature s except that u is a function of the view (ϕ, θ) , the 3D angle β_w (instead of α_w) and the 3D length ratio t_w (instead of s_w), for β_w and t_w defined in 5.1. Without loss of generality, this function can be represented by considering a 3D line segment configuration equivalent to the one used for s , except that \vec{P}_3 is in the position of \vec{P}_4 , and the relevant angle and length are β_w and t_w . By considering the projections of this configuration, we get the following expression for u :

$$\begin{aligned} u &= \vec{p}_3 \vec{p}_2 / |\vec{p}_2|^2, \\ \vec{p}_2 &= \pi R_{\phi, \theta}(\cos(\beta_w/2), -\sin(\beta_w/2), 0)^T \\ \vec{p}_3 &= t_w \pi R_{\phi, \theta}(\cos(\beta_w/2), \sin(\beta_w/2), 0)^T \end{aligned}$$

As in the case for s , u is a linear function of the 3D distance ratio t_w and the following observations can be made for the case of u :

- When β_w is zero (i.e., the 3D points P_1, P_2 and P_3 are collinear), u is constant (t_w) for all (θ, ϕ) . This is analogous to the special invariant called the *approach ratio* in [Brooks81].
- As β_w approaches zero, the variation as a function of (θ, ϕ) is slow over most of the views.

⁹In this way, the length ratio $s = |\vec{p}_4|/|\vec{p}_2|$ has the same variational behavior as its inverse, which seems reasonable.

- When $\beta_w > 0$, there always exists some view where $u = \infty$ and $u = -\infty$. In other words, u covers the full range of possible values. The value u approaches ∞ when $\theta = 90$ and ϕ approaches $-\beta_w/2$ from a counter-clockwise direction, and it approaches $-\infty$ when $\theta = 90$ and ϕ approaches $-\beta_w/2$ from a clockwise direction (the view is lining up with $(P_2 - P_1)$ from different directions). It approaches zero when (θ, ϕ) approach $(90, \beta_w/2)$ (this is when the view lines up with the displacement vector $(P_3 - P_1)$).
- As with the projected features already discussed, u is slowest in variation when $\theta = 0$ (i.e., the view is most normal to the plane containing the points P_1, P_2 and P_3). At this point, $u = t_w \cos \beta_w$. Also, variation in u is still fairly slow for views near the Z axis, especially if the 3D angle β_w is reasonably small.

Figure 11 shows the histograms of the number of views that fall within regular intervals of u for various 3D angles β_w , with t_w set to one and the same view sampling scheme as in Section 5.3. The projected size ratio u actually ranges from minus infinitely to plus infinity, but for these examples, the histograms range over $[-2, 2]$ and contain almost all of the samples.

As the histograms show, the distributions of the projected features are strongly concentrated about the value of the true 3D feature $u_w = t_w \cos \beta_w$, and this concentration becomes more pronounced as β_w approaches zero (similarly for 180 degrees). A more revealing picture can be gained by counting the number of views for which u falls within some interval about the true feature, $[u_w + \Delta u, u_w - \Delta u]$.

Figure 12 shows the percentage of the view-sphere that falls within this interval for various Δu ranging from 0 to 2 and β_w of 90, 45, 22.5 and 11.25 degrees. The amount of the view sphere that falls within an interval clearly gets larger as β_w decreases. For the values of β_w studied, a smallish interval of $[-0.2, 0.2]$ covers approximately 42, 52, 70 and 87 percent of the sphere respectively. To guarantee that most of the sphere (say 80 percent of it) falls within the interval, Δu has to be approximately .85, .58, .3 and .15 for each of the β_w respectively.

6 Summary and conclusions

Though it has been shown that there are no general-case view-invariants, there are special-case invariants of practical importance. For weak perspective, the special cases can be understood in terms of linear dependence and the invariance of this relation to linear transformation.

Also for weak perspective, there are 2D features with value distributions that tend to be restricted enough in the general case to provide useful features for 3D object recognition. The relative features of the projection, α , s , u and v are functions of two view parameters (θ, ϕ) and one or two 3D structure parameters (one for α and two for the others). All of these features vary slowest when θ is near zero and at this view assume the corresponding 3D features $\alpha_w, s_w, u_w = t_w \cos \beta_w$ and $v_w = t_w \sin \beta_w$. When the directions or displacements

are parallel ($\alpha_w = 0$ or $\beta_w = 0$) each feature becomes invariant to view. When they are non-parallel, each feature always varies across its whole range of possible values. In the case of s, u and v , the values "blow up" (i.e., assume their extreme values) when the view aligns with a 3D displacement of importance to the feature. In the case of α , extreme values occur when the view lines up with the 3D angle bisector or a position in the object plane normal to it. In spite of the fact that the projected features vary across the full range of their values for most 3D line segment configurations, all of the features seem to vary reasonably slowly for a usable range of views positioned away from the planes containing the object points; i.e., $\theta \ll 90$. This is especially true for smaller 3D angles (α_w or β_w).

7 References

- Ahuja, D.V. and S.A. Coons, "Geometry for Construction and Display", IBM Systems Journal, 7(3-4), pp. 188-205, 1968.
- Aliomonos, Y. and M. Swain, "Paraperpective Projection: Between Orthography and Perspective", CAR-TR-320, U. Maryland, 1987.
- Binford, T.O., T.S. Levitt, W.B. Mann, "Bayesian Inference in Model-Based Machine Vision", Proc. AAAI Uncert. Work., 1987.
- Brooks, R.A., "Symbolic Reasoning among 3D Models and 2D Images", AI, vol. 17, pp. 285-348, 1981.
- Burns, J.B. and L.J. Kitchen, "Recognition in 2D images of 3D Objects from Large Model Bases using Prediction Hierarchies", Proc. IJCAI-10, 1987.
- Burns, J.B., R. Weiss and E.M. Riseman, "Analysis of View Variation of Point Set and Line Segment Features", COINS TR, in prep., UMass, Amherst, 1990.
- Duda, R. and P. Hart, *Pattern Classification and Scene Analysis*, Wiley-Interscience, 1973.
- Grimson, W.E.L. and T. Lozano-Perez, "Model-based Recognition and Localization from Sparse Range and Tacite Data", IJRR, vol 3, pp. 3-35, 1984.
- Ikeuchi, K. "Precompiling a Geometrical Model into an Interpretation Tree for Object Recognition in Bin-picking Tasks", DARPA IUW, pp. 321-339, 1987.
- Horn, B.K.P., "Robot Vision", MIT Press, 1986.
- Huttenlocher, D.P. and S. Ullman, "Object Recognition using Alignment", Proc. ICCV, pp. 102-111, 1987.
- Korn, M.R. and C.R. Dyer, "3D Multi-view Object Representations for Model-based Object Recognition", Pattern Recognition, vol. 20(1), pp. 91-103, 1987.
- Lamdan, Y., J.T. Schwartz and H.J. Wolfson, "Object Recognition by Affine Invariant Matching", Proc. CVPR, pp. 335-344, 1988.
- Lamdan, Y., H.J. Wolfson, "Geometric Hashing: A General and Efficient Model-Based Recognition Scheme", Proc. CVPR, pp. 238-249, 1988.
- Lowe, D.G., *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, 1985.
- Marr, D. *Vision*, W.H. Freeman, 1982.
- Thompson, D.W. and J.L. Mundy, "Three-dimensional Model Matching from an Unconstrained Viewpoint", Proc. IEEE Int. Conf. on Rob. and Auto., pp. 208-220, 1987.
- Tucker, L.W., C.R. Feynman, D.M. Fritzsche, "Object Recognition using the Connection Machine", Proc. CVPR, 1988.
- Weiss, I., "Projective Invariants of Shapes", DARPA IUW, pp. 1125-1134, 1988.

POSE REFINEMENT: APPLICATION TO MODEL EXTENSION AND SENSITIVITY TO CAMERA PARAMETERS

Rakesh Kumar and Allen R. Hanson

Computer and Information Science Department
University of Massachusetts at Amherst *

Abstract

In this paper, we study the effect of errors in estimates of the image center and focal length on pose refinement and other related (3D inference from 2D images) problems/algorithms. The goal in pose refinement is to find the rotation and translation (or location) matrices which map the world coordinate system to the camera coordinate system. We show that for "small" field of view imaging systems, incorrect knowledge of the camera center does not affect the location of the camera significantly. The rotation is affected however, and the amount of error in the rotation is linearly related to the incorrect estimate of the center. Finally, it is shown that incorrect estimates of the focal length only significantly affects the z-component (i.e. parallel to the optical axis) of the translation in camera coordinates.

The output of the pose refinement algorithm is used to calculate the relative orientation between the coordinate frames of the same camera in two different positions as a prelude to computation of 3D depths of new points by triangulation. A model of error for this depth is constructed based on the amount of error in placing the image center. The errors predicted by this model conform to the errors obtained for experiments with synthetic and real data. The induced stereo process is extended to multiple frames to make robust estimates of the 3D locations of new points. This process is called Model Extension. Results are presented for two real image sequences. New points are located to an average accuracy of 1.5mm and 0.3 feet for the two sequences respectively.

1 Introduction

The standard model adopted for imaging 3D scenes by CCD and other cameras is perspective projection. A ray from the camera focal point to a 3D point intersects the image plane at the image location of the 3D point under perspective projection. The optical axis is defined as the perpendicular line from the focal point to the imaging plane and the image center is defined as the point where the optical axis pierces the image plane.

Two important camera parameters which often need to be calibrated are the focal length and the image center. In this paper, we study the effect of errors in estimates of the image center and focal length on pose refinement and other related (3D inference from 2D images) problems/algorithms. The pose refinement algorithms used in the experiments are described in [6]. The conclusions drawn, however, are independent of the particular algorithm used.

The image center is often assumed to lie at the center of the image frame. This default center has been reported to be off by as much as 30 pixels for some standard camera and frame grabber combinations [8]. Calibration techniques using either lasers or high precision calibration plates have been used to locate the center to within a few pixels [4,8,10]. Is this precise calibration necessary? The analysis presented here shows that it depends on three factors:

1. The particular 3D output or inference one is interested in.
2. The level of accuracy desired in the results.
3. The amount of noise in the input data.

The goal in pose refinement is to find the rotation and translation matrices which map the world coordinate system to the camera coordinate system. Given the rotation (or orientation) and translation, the location of the camera with respect to the world coordinate system can be computed. We will show that for *small field of view* imaging systems, an error in the estimation of the camera center does not affect the location of the camera significantly. The rotation or orientation is affected, however, and the amount of error in the orientation is linearly related to the error in the estimate of the center.

An application of the pose refinement process is model extension. Given a partial model of the scene, it can be used to obtain robust 3D estimates of new image features, effectively extending the model. From the poses computed using the partial model for two images taken from the same camera, the relative orientation between the two image coordinate frames is computed as a prelude to "induced stereo" analysis¹. Using the computed

*This research was supported by the following Defense Advanced Research Projects Agency grants F30602-87-C-0140, DACA76-89-C-0017 and National Science Foundation grant DCR8500332.

¹We use the term "induced stereo" to refer to the process of estimating 3D locations of points from triangulation given the relative orientation between the same camera in

relative orientation, the 3D depth and location of points (in the coordinate frame of one of the cameras) is computed using triangulation. In the third section of this paper a model of error for this depth is constructed based on the amount of error in locating the image center. The errors predicted by this model are consistent with the errors obtained when applying the pose refinement algorithm [6] to both synthetic and real data. We show that these errors are small compared to the errors caused by image noise of up to 0.5 pixels for 512×512 images with 24 deg. field of view and 3 feet long stereo baselines². Furthermore, if the 3D coordinates of the triangulated point are transformed to the world coordinates using the computed pose, the error in 3D location is only due to second order effects and hence negligible for small field of view systems. In section 4, the induced stereo process is extended to multiple frames. Image tokens are tracked over a sequence of frames using the computed optic flow between pairs of successive frames. Results are presented for two real image sequences. New points are located to an average accuracy of 1.5mm and 0.3 feet for the two sequences respectively.

The results derived for induced stereo, showing the effect of errors in locating the image center on the relative orientation between pairs of frames, are also applicable to recovery of structure from motion algorithms [5]. Experiments for motion in depth show that these formulae were able to predict moderately well changes in relative orientation³ as computed by Horn's algorithm [5]. However, the formulae did not predict the errors well for experiments with motion parallel to the image plane. In the case of induced stereo, the formulae were accurate in their predictions for both kinds of motions. Note that structure from motion algorithms are especially non-robust when the motion is parallel to the image plane [1].

Finally, in the last section of this paper, the effect of incorrect estimation of the focal length on the pose refinement problem is studied. We show that incorrect estimates of the focal length only significantly affects the z-component (i.e. parallel to the optical axis) of the translation. The x and y components of the translation and the rotation are not affected significantly. However, the location of the camera in world coordinates will be affected since the z-component of the translation changes. Again, experimental results on real data are presented to support the theoretical claims.

2 Errors in the pose refinement problem from center offsets

The question asked is this: given two input data sets to the pose refinement problem (the first with the correct image center and the second with an offset image center) how are the two resulting poses related? The only difference between the two data sets is a constant offset

two different locations.

²Therefore image noise is the most significant factor in determining accurate 3D depths.

³Due to errors in locating the image center.

of all the image pixels in one data set by the amount the center estimate is offset. Associated with each of the input data sets is a camera coordinate frame. The result of the pose refinement process is to determine the rigid body transformation between the world coordinate frame and the camera coordinate frame. Let "W" represent the world coordinate frame, "C1" the camera coordinate frame with the correct center and "O1" the camera coordinate frame with the offset center; then

$$X_{c1} = R_{c1}(X_w) + T_{c1} \quad (1)$$

In this equation, the rotation R_{c1} and translation T_{c1} relate a 3D point X_{c1} in the first camera coordinate frame "C1" to its coordinates X_w in the world coordinate frame "W". Points in the camera coordinate frame "O1" are related to points in the world coordinate frame "W" by equation:

$$X_{o1} = R_{o1}(X_w) + T_{o1} \quad (2)$$

We would like to find the relationship between the two camera coordinate frames "C1" and "O1". As noted earlier the only difference between the image data associated with the two frames is a constant shift of all the pixels. Let these be ΔC_x and ΔC_y in the X and Y image frame directions, respectively; these shifts correspond to the offset of the image center for the second data set. The displacement of image points between two frames due to rigid motion [2] is given by the following equation:

$$\alpha = \frac{x_1 y \Omega_x}{f} - (f + \frac{x x_1}{f}) \Omega_y + y \Omega_z + \frac{(f T_z - x T_x)}{Z} \quad (3)$$

$$\beta = (f + \frac{y y_1}{f}) \Omega_x - \frac{y_1 x \Omega_y}{f} - x \Omega_z + \frac{(f T_y - y T_z)}{Z} \quad (4)$$

where

α, β are the image displacements in the x, y axis respectively.

$(\Omega_x, \Omega_y, \Omega_z)$ are the small angle approximations to rotation about the X, Y and Z axis respectively.

(T_x, T_y, T_z) is the translation along the (X, Y, Z) axis respectively.

Z is the depth of the point in the first coordinate frame.

f is the focal length of the camera in pixels.

(x, y) is the location of the point in the first image frame ("C1") and (x_1, y_1) is the location of the point in the second image frame ("O1").

Between the two frames "C1" and "O1", $\alpha = \Delta C_x$ and $\beta = \Delta C_y$ i.e. both are constant for all points in the image. What transformation can account for this constant shift? If we assume the field of view of the camera is small, then second order terms such as $x x_1$, $x_1 y$ etc. can be neglected. If the scene being imaged is not a frontal plane, i.e. "Z" is not constant for all points⁴ then the only transformation that can cause a constant change for a general set of points is the rotations Ω_x and Ω_y about the X and Y axis; everything else

⁴Frontal planes are dealt with later on.

(i.e. Ω_x , T_x , T_y and T_z) will be zero. The following two equations express this relationship:

$$\alpha = \Delta C_x = -f\Omega_y \quad (5)$$

$$\beta = \Delta C_y = f\Omega_x \quad (6)$$

Let the rotation operator Δ_R represent the overall rotation composed of the rotations Ω_x and Ω_y about the X and Y axis. The two coordinate frames "C1" and "O1" are therefore hypothesized to be related by a rotation Δ_R :

$$X_{o1} = \Delta_R(X_{c1}) \quad (7)$$

Combining equation (7) with equation (1) we get:

$$X_{o1} = \Delta_R R_{c1}(X_w) + \Delta_R(T_{c1}) \quad (8)$$

Comparing equation (8) with equation (2) we see that:

$$R_{o1} = \Delta_R R_{c1} \quad (9)$$

$$T_{o1} = \Delta_R(T_{c1}) \quad (10)$$

The above equations reflect how the orientation R_{o1} and location of the world origin in camera coordinates T_{o1} are altered with incorrect knowledge of the center. The location of the camera origin in world coordinates T_w is given by the following equation:

$$T_{wc1} = -R_{c1}^T(T_{c1}) \quad \text{for camera frame C1.} \quad (11)$$

$$T_{wo1} = -R_{o1}^T(T_{o1}) \quad \text{for camera frame O1.} \quad (12)$$

Using equations (9, 10) and the above equation for T_{wo1} we get:

$$T_{wo1} = -R_{c1}^T \Delta_R^T \Delta_R(T_{c1}) = -R_{c1}^T(T_{c1}) = T_{wc1} \quad (13)$$

Therefore an error in estimating the image center does not affect the location of the camera in world coordinates significantly. It is only affected if the second order terms in the motion displacement equations (3,4) are significant. For small field of view imaging systems, they are not significant. However, the orientation of the robot is affected; the amount it is affected depends on the values of $(\Delta C_x, \Delta C_y)$. For instance, for a camera with field of view 24 deg. and a 512 x 512 image, a 30 pixel offset in the camera center in either x or y coordinate would cause a rotation error of 1.427 deg. about the corresponding axis. Whether changes in orientation of this order are significant or not depends on the application.

Finally, in the case of frontal planes, the depth value "Z" is the same for all points. Therefore in the motion displacement equations (3,4) both the translation components T_x , T_y and rotation terms Ω_x , Ω_y can account for the constant displacement. In this case, the model of change in pose as given in equation (7) may not be correct. However, the reader is reminded that frontal planes are typically a degenerate case for pose. Even if we have a correct estimate of center, since "Z" is constant, there could be an incorrect pose related to the correct pose by a transformation composed of translation components T_x , T_y and rotation components Ω_x , Ω_y . The image transformations caused by rotation (Ω_x , Ω_y) can be cancelled by the transformation due to translation (T_x , T_y) in equations (3,4) leading to approximately zero values of

α and β and therefore more than one pose can explain the same input data. The same observation has been made for the structure from motion problem by other researchers [7]. The above model will also break down for large field of view imaging systems (e.g. beyond 45 deg. field of view), i.e. when the second order effects cannot be ignored.

2.1 Experimental Results

In an earlier paper we described algorithms for pose estimation given 3D model - 2D image point and line correspondences [6]. We show results from our pose algorithms for two image sets with different errors in the locating the image center. The images (512 x 484 pixels) were acquired using a SONY B/W camera (model AVC-D1) interfaced to a GOULD frame grabber. The field of view of the imaging system is approximately 24.0 degrees. For each set of image data, a new data set was created by adding a constant pixel offset to the x and y coordinates of the image data of the original set.

The first image (Fig. 1) is of a hallway; the door in the image is 40 feet distant from the camera. Fig. 1 shows the first set of input image lines to the pose algorithm. Two more sets of input data were created by adding center offsets of 10 and 20 pixels respectively. Fig. 2 shows the projected lines after estimation of pose for the first (original) set of input image data. Fig. 3 shows the projected lines after estimation of pose for the third set (center offset of 20 pixels) of input image data. Note that to display the data in Fig. 3, the original intensity image was shifted by 20 pixels on each axis (corresponding to the center offset). It is clear from Fig. 2 and Fig. 3 that the projections align with their respective input images in a very similar manner. The results for location of the camera in world coordinates for the three different center offsets is given in Table 1 under the heading "HALLWAY IMAGE". The final location (in feet) in world coordinates (for scenes and images as shown in the figure above) changes only by a few tenths of an inch. The (0,0) offset corresponds to the projected model in Fig. 2 and the (20,20) offset corresponds to the projected model in Fig. 3.

The second image is shown in Fig. 4. The camera was about 650 mm distant from the top corner of the box. The fifteen points marked by crosses in Fig. 4 were provided as input to the pose refinement algorithm. Three new image data sets were created by adding center offsets of (10,0), (10,10) and (20,20) respectively. The results of locating the camera for these different data sets are shown in Table 1 under the heading "BOX IMAGE". As can be seen from the table the location of the camera changes by only 1 or 2 mm for different center offsets. Although results from only two images are presented here, the above behaviour has been observed for numerous other images.

3 Errors in induced stereo from center offsets

Given two image frames from the same camera at two different positions, the relative orientation between the

Table 1: Location of camera in world coordinates as computed by the pose refinement algorithm for two sets of real image data with different center offsets.

Center Offset X	Center Offset Y	LOCATION in WORLD		
		L_x	L_y	L_z
HALLWAY IMAGE				
pixels	pixels	feet	feet	feet
Measured	Location	40.00	4.00	3.57
0	0	39.98	4.09	3.57
10	10	40.00	4.09	3.58
20	20	40.02	4.09	3.58
BOX IMAGE				
pixels	pixels	mm	mm	mm
0	0	418.23	260.52	381.37
10	0	417.94	260.49	381.72
10	10	417.27	260.56	380.85
20	20	416.68	260.71	380.51

camera coordinate systems for the two frames can be computed using a pose recovery algorithm. The relationship of the two cameras with respect to the world coordinate system is found and from that the relative orientation is computed. Let the two frames with the correct center be "C1" and "C2"; their relationship to the world coordinate system is:

$$X_{c1} = R_{c1}(X_w) + T_{c1} \quad (14)$$

$$X_{c2} = R_{c2}(X_w) + T_{c2} \quad (15)$$

Combining these equations, the relative orientation between the frames "C1" and "C2" can be expressed by:

$$X_{c1} = R_{c12}(X_{c2}) + T_{c12} \quad (16)$$

$$R_{c12} = R_{c1}R_{c2}^T \quad (17)$$

$$T_{c12} = (T_{c1} - R_{c1}R_{c2}^T(T_{c2})) \quad (18)$$

Similarly, let the frames with the incorrect center be "O1" and "O2". The relative orientation between these frames is given by:

$$X_{o1} = R_{o12}(X_{o2}) + T_{o12} \quad (19)$$

$$R_{o12} = R_{o1}R_{o2}^T \quad (20)$$

$$T_{o12} = (T_{o1} - R_{o1}R_{o2}^T(T_{o2})) \quad (21)$$

Using equations (9,10) and some algebraic manipulation, we can rewrite equations (21) and (21) for R_{o12} and T_{o12} in terms of R_{c12} , T_{c12} and Δ_R :

$$R_{o12} = \Delta_R R_{c12} \Delta_R^T \quad (22)$$

$$T_{o12} = \Delta_R(T_{c12}) \quad (23)$$

The above two equations represent the error in the relative orientation if the center estimate is incorrect. If two corresponding points in the two frames "C1" and "C2" are given and assuming the unit vectors⁵ from the focal

point to them are r_{c1} and r_{c2} respectively, the formula for depth D_c of the 3D point obtained by triangulation in the first camera coordinate frame is:

$$D_c = s(r_{c1} \cdot \hat{z}) \quad (24)$$

$$s = \frac{(r_{c1} \times R_{c12}(r_{c2})) \cdot (T_{c12} \times R_{c12}(r_{c2}))}{\| (r_{c1} \times R_{c12}(r_{c2})) \|^2} \quad (25)$$

In this equation s is the length along the 3D ray corresponding to the vector r_{c1} from the origin of the 3D point and \hat{z} is the unit vector along the z-axis.

For the frames "O1" and "O2" the unit vectors for the same points r_{o1} and r_{o2} corresponding to points r_{c1} and r_{c2} in frames "C1" and "C2" can be approximated as before⁶ by:

$$r_{o1} = \Delta_R(r_{c1}) \quad (26)$$

$$r_{o2} = \Delta_R(r_{c2}) \quad (27)$$

Combining these two equations and equation (24), the depth of the same 3D point in the image frame "O1" coordinate system is:

$$D_o = s(r_{o1} \cdot \hat{z}) = s(\Delta_R(r_{c1}) \cdot \hat{z}) \quad (28)$$

Note that the length along the 3D ray has not changed in the offset center case, i.e. frame "O1-O2" as compared to the correct center pair "C1-C2". However, the depth of the point changes because of the rotation of the unit vector r_{c1} . The X and Y coordinates of the 3D point are also similarly affected.

From the above derivation the percentage error in depth can be predicted by the following formula:

$$\%D_{err} = \frac{-\Delta C_y r_{c1x} + \Delta C_x r_{c1y}}{f r_{c1z}} 100.0\% \quad (29)$$

We use this error to predict the percentage depth error due to incorrect center and compare it with the actual depth errors found when running pose and the triangulation algorithm on synthetic data. We also compare the errors in depth computation due to an incorrect estimate of center and noise in the image locations versus data with a correct estimate of center but no noise being present. As results show, the error for even small amounts of image noise are much larger than error due to incorrect center placement.

Note if the triangulated 3D point is transformed to world coordinates, then the only error will be due to second order effects. The center offset causes the 3D point to be rotated by Δ_R in the camera coordinate system and the subsequent transformation back to world coordinates cancels out the Δ_R rotation.

3.1 Experimental results for induced stereo

Experimental results are presented in this section for synthetic data and real image data. A pair of images is required to do each experiment for this section. Synthetic data was created by taking a model of a 3D scene very similar to the hallway shown in Fig. 1 and projecting the 3D points onto to the image plane for two

⁵We define the rays corresponding to these vectors as projection rays.

⁶The approximation is ignoring the second order terms for small field of view systems.

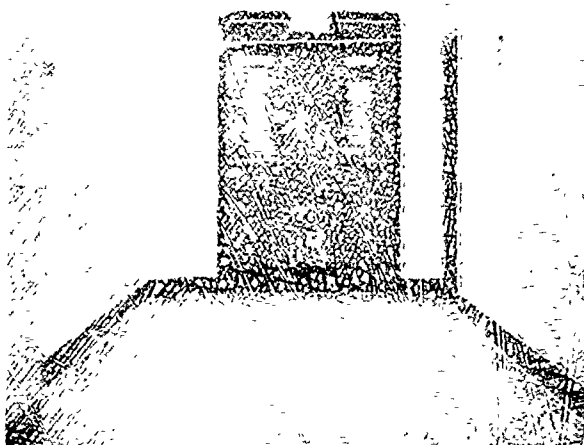


Fig. 1: 24 input data lines to pose algorithm. 512 x 512 image with field of view equal to 24 deg..

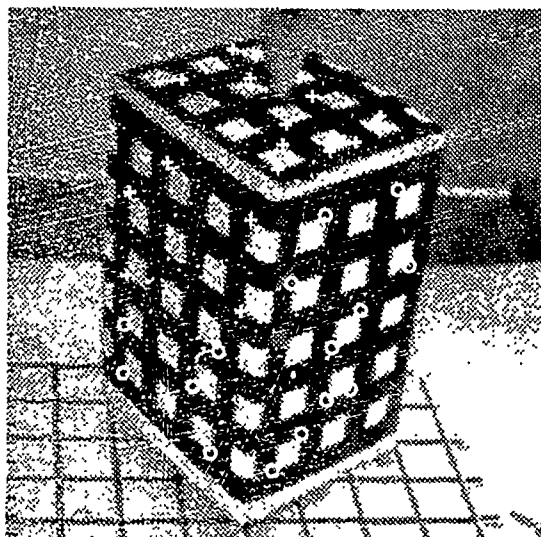


Fig. 4: Box Image. Points marked by crosses used for pose. Points marked by circles use for depth estimation.

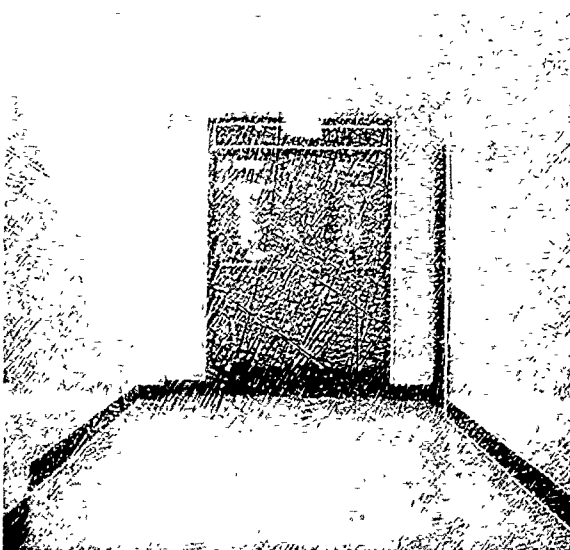


Fig. 2: Projected lines after estimation of pose, image center assumed to be frame center; input data lines are shown in Fig. 1.

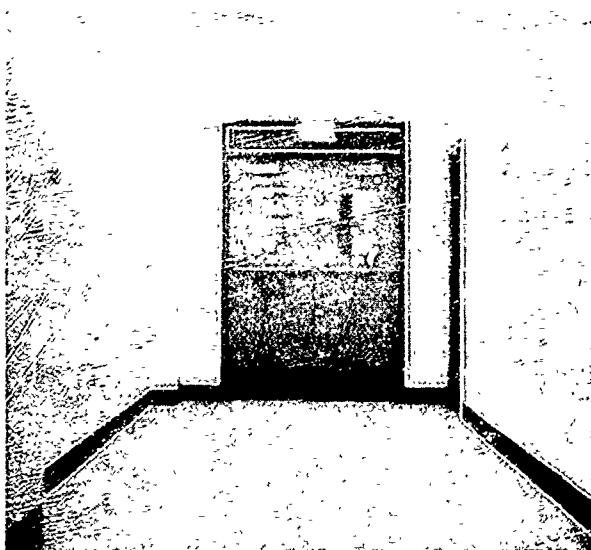


Fig. 3: Projected lines after estimation of pose, image center assumed to be offset from frame center by 20 pixels along each axis. Note, to display this figure, the original intensity image shown in Fig. 1 was shifted by 20 pixels along each axis.

different positions of the camera (induced stereo baseline was approximately 3.0 feet). Twenty points in total were used, out of which only 9 were used for pose calculation. Depth computations were done for all twenty points. The imaging frame was assumed to be 512 x 512 with a field of view equal to 24 deg..

The box image shown in Fig. 4 was the first frame used for the real image data. The second image was obtained by rotating the box by approximately 25 degrees about its central vertical axis. The fifteen points marked by crosses in Fig. 4 were used to compute the pose for each frame. Depths were computed for the fifteen points marked by circles in Fig. 4.

Table 2: Predicted percentage average depth errors versus computed average depth errors for different center offsets for synthetic and real image data.

Center Offset X pixels	Center Offset Y pixels	Predicted % depth error	Computed % depth error
SYNTHETIC IMAGE			
10	0	0.063	0.047
10	10	0.085	0.091
20	20	0.169	0.183
30	30	0.254	0.277
50	50	0.423	0.469
BOX IMAGE			
10	0	0.082	0.078
10	10	0.141	0.172
20	20	0.283	0.337
30	30	0.424	0.495
50	50	0.707	0.789

In Table 2 we compare the predicted percentage average depth errors versus the computed average depth errors for various different center offsets for both the synthetic data and the box data. As can be seen, the predicted depth errors compare quite favorably to the computed ones. The very small difference between the predicted and computed errors can be attributed to the second order effects which were ignored.

For the comparison of error due to incorrect center versus error due to noisy image locations, we added various amounts of uniform pixel noise to the synthetic image data. The center was offset by a constant amount of 30 pixels in each axis for this experiment. From Table 3, it can be seen that at noise levels of greater than 0.5 pixels, the error with and without center offset are comparable. It is only at image noise levels of less than 0.5 pixels that the error due to incorrect center is significant. Of course, if we increase the induced stereo baseline and are able to make more accurate 3D measurements from induced stereo, then the 3D error caused by incorrect center estimates will become comparable to 3D errors at larger levels of image noise. To conclude, given a particular stereo configuration and expectation of image noise, we can calculate the significance of 3D

Table 3: Computed average depth errors for synthetic uniform noise data with and without center offsets. 512 x 512 image with 24 deg field of view, center offset by 30 pixels for each axis, 3 feet long stereo baseline.

Image Noise pixels	Noise only % Depth Err	Center Offset plus Noise % Depth Error
0.0	0.000	0.277
0.1	0.124	0.300
0.2	0.247	0.350
0.5	0.623	0.661
1.0	1.366	1.432
1.5	1.453	1.424
2.0	2.133	2.240
3.0	3.398	3.418
5.0	5.653	5.676
10.0	11.638	11.765

error due to an error in estimating the center.

3.2 Structure from Motion

The equations which show the effect of errors in locating the image center on the relative orientation between pairs of frames, derived in the case of induced stereo are also applicable to recovery of structure from motion algorithms. The error function E_h minimized by Horn [5] in his relative orientation algorithm given point correspondences for a pair of frames is:

$$E_h = \sum_{i=1}^n ((r_{c1i} \times R_{c12}(r_{c2i})) \cdot T_{c12})^2 \quad (30)$$

where

r_{c1i} , r_{c2i} are the vector representations of the projection rays of corresponding points.

R_{c12} and T_{c12} are the relative orientation parameters: rotation and translation respectively.

If we create two new frames, by shifting the original image data by an offset corresponding to the error in locating the center, the error function E_h^o to be minimized is:

$$E_h^o = \sum_{i=1}^n ((r_{o1i} \times R_{o12}(r_{o2i})) \cdot T_{o12})^2 \quad (31)$$

Substituting in equation (31) for the new projection rays (r_{o1i} , r_{o2i}) using equations (26) and (27) and for the rotation R_{o12} and translation T_{o12} using equations (22) and (23) respectively, we can show that:

$$E_h^o = E_h \quad (32)$$

Therefore, if E_h is minimum for the rotation R_{c12} and translation T_{c12} then E_h^o is minimum for the rotation R_{o12} and translation T_{o12} which are related to (R_{c12} , T_{c12}) by equations (22) and (23). The change in relative orientation caused by errors in estimating the center are predicted by equations (22) and (23).

Experimentally, these formulae were able to predict moderately well changes in relative orientation⁷ computed by Horn's algorithm [5] for motions in depth but not at all accurately for motions parallel to the image plane. Note that structure from motion algorithms are especially non-robust when the motion is parallel to the image plane.

4 Model Extension

An application of the pose refinement process is model extension. Given a partial model of the scene, it can be used to obtain robust 3D estimates of new image features, effectively extending the model. In the previous section, we discussed how this can be done using triangulation and two image frames. However, two image frames provide only one 3D measurement of the point. To increase the robustness of the computations, 2D information from a sequence of frames is combined. In this section, we present techniques for computing 3D estimates of new points in the world coordinate system using a sequence of frames and a partial 3D model of the scene being imaged.

Image features (both new features and modelled image features appearing in the images) are tracked over a sequence of frames using the computed optic flow between pairs of successive frames [11]. Typically we track corners (defined by the intersection of two image lines) although any image feature which can be reliably tracked may be used. The initial matching of image lines to the partial model for the first frame may be done by a matching process such as in [3]. Combining the results of the initial matching and the feature tracking, correspondences between image features and the partial model for each frame are established. Using these correspondences, pose estimation is done for each frame. The image projection ray for an image point for a particular frame is defined as the ray originating from that frame's optic center and passing through the image point. Given the pose estimates for each frame, the vectors corresponding to these projection rays in the world coordinate system can be obtained. The 3D estimate of the point is the pseudo-intersection of all the image projection rays for a tracked image point. A nice property of this system is that in order to combine 3D measurements from a sequence of frames, a stable coordinate frame should be used; the pose estimation process provides the world coordinate system as this stable coordinate system. Independent measurements can be made relating the coordinate system of each frame in the sequence to the world coordinate frame.

We now describe how the pseudo-intersection is done. Let r_i be the unit vector corresponding to the image projection ray for an image point in the i 'th frame. The pose estimation for this frame is given by the rotation R_{ci} and translation T_{ci} (see equation (1)). We wish to find the 3D point X_w in world coordinates which is the pseudo-intersection of all the image projection rays for the tracked image point over the entire sequence.

Since the image projection rays do not intersect at a unique point, an optimization procedure is used to minimize the sum of squares of the perpendicular distances from the 3D pseudo-intersection point to the image projection rays. The error function E minimized is:

$$E = \sum_{i=1}^n \|(R_{ci}(X_w) + T_{ci}) \times r_i\|^2 \quad (33)$$

Using elementary vector algebra, we can show that E is:

$$E = \sum_{i=1}^n (\|R_{ci}(X_w) + T_{ci}\|^2 - ((R_{ci}(X_w) + T_{ci}) \cdot r_i)^2) \quad (34)$$

In this equation, the unknown variable is the 3D point X_w in world coordinates. Differentiating E with respect to X_w and setting the resulting expression equal to zero results in a set of linear equations in X_w :

$$nX_w - \sum_{i=1}^n (X_w \cdot r_i') r_i' = - \sum_{i=1}^n a_i \quad (35)$$

where

$$r_i' = R_{ci}^T(r_i)$$

$$a_i = R_{ci}^T(T_{ci}) - (T_{ci} \cdot r_i') r_i'$$

Thus the algorithm for model extension can be summarized as follows:

- Step 1 Given a partial 3D model and an image, establish correspondences between model points and image points using a matching technique such as in [3].
- Step 2 Track image points over a sequence of frames using the computed optic flow between successive pairs of images [11].
- Step 3 Using the correspondences established above between model points and image points, compute the pose [6] for each image frame.
- Step 4 Estimate the 3D location of a new point in world coordinates using the linear system of equations (35) and the feature correspondences established in Step 2.

4.1 Experimental Results

This algorithm has been applied to two image sequences. Fig. 4 and Fig. 5 show the images of the 1'st and 14'th frame in the BOX and PUMA sequences respectively. In both experiments the image center was assumed to be at the center of the image frame and the effective focal length was calculated from manufacturers spec. sheets. Calibration for intrinsic camera parameters has not been done.

The first sequence (referred to as the BOX sequence) was generated by rotating the box (in Fig. 4) about its central vertical axis, the camera being kept stationary. Consecutive images in the sequence were taken after a rotation of approximately 3.6 degree [9]. The camera was about 650 mm distant from the top front corner of the box. The location of 30 points (marked in Fig. 4) in a world coordinate system was measured to an accuracy of approximately 1 mm along each axis. The depth of

⁷Due to errors in locating the image center.

the points (in the first frame's coordinate system) used in our experiment varied from 575 mm to 700 mm. The thirty points were tracked over the set of 8 frames. The fifteen points marked by crosses in Fig. 4 were used to do pose estimation [6] for each frame. Computed 3D estimates of the remaining 15 points (marked by circles in Fig. 4). were compared with measured 3D locations for these points. The average error in the computation was 1.42 mm. The maximum error was 2.16 mm and the minimum error was 0.48 mm. The average percentage error was 0.25 %. The percentage error is calculated by dividing the absolute 3D error by the depth of the point from the origin of the camera in the first image's coordinate frame.

In this experiment, the high accuracy with which 3D parameters of the new points were computed is due primarily to the fact that the motion over the sequence is approximately parallel to the image plane. Such motion is best for accurate triangulation. Moreover, due to the rotation about an off-centered axis, image features remain in the image plane for the entire sequence and large image disparities are obtained. Unfortunately, similar results are not obtained when the motion of the camera is mostly in depth. In this case, the displacement of image points near the FOE is small and not many points remain visible for a large number of frames.

In the first experiment described above for the box sequence, the image center was assumed to be at the frame center. In another experiment, the image center was assumed to be displaced by 15 pixels along each axis from the frame center. The experiment was repeated and the 3D locations of the points obtained; comparing these locations to the previously computed locations, we found that the new estimates of the 3D points were off from the previously computed estimates by an average distance of 0.261 mm. This supports the earlier claim that incorrect estimates of the center do not affect the 3D estimation of points significantly for small field of view systems (24 deg. for this sequence).

The second sequence was generated by fixing a camera to a PUMA arm and rotating the arm by 4 degrees between consecutive positions of the camera. The field of view of the imaging system was 40 degrees. Fig. 5 shows the 14'th frame of this sequence (referred to as the PUMA sequence). The plane of rotation of the camera is approximately parallel to the image plane. The axis (off-centered) of rotation intersects the image plane somewhere between points 8 and 18 in Fig. 5. The radius of rotation is approximately 2 feet. Thirty frames were taken and the total angular displacement is 116 degrees. The maximum displacement of the camera in these thirty frames is approximately 2 feet along the world y-axis (vertical direction) and 1 foot along the world x-axis (parallel to the x-axis of the image in Fig. 5). This corresponds to the longest baseline over these 30 frames. The location of 32 points (marked in Fig. 5) in a world coordinate system was measured to an accuracy of approximately 0.2 feet along each axis. The depth of the points (in the first frame's coordinate system) used in our experiment varied from 13 feet to 33 feet. Most of the

32 points were tracked over the entire set of 30 frames.

Table 4: Absolute and Percentage 3D location errors for points in PUMA sequence (see Fig. 5.)

Point Num.	Depth feet	Absolute Error feet	Percentage Error
1	24.59	0.616	2.50 %
2	26.02	0.355	1.36 %
3	28.32	0.373	1.32 %
4	22.06	0.440	1.99 %
5	30.20	0.217	0.72 %
6	28.62	0.281	0.98 %
7	31.56	0.472	1.50 %
8	32.61	0.038	0.12 %
9	14.33	0.125	0.87 %
10	15.34	0.279	1.82 %
11	14.46	0.019	0.13 %
12	13.50	0.081	0.60 %
13	21.75	0.054	0.25 %
14	18.81	0.022	0.12 %
15	21.73	0.036	0.17 %
16	20.28	0.104	0.51 %
17	21.26	0.402	1.89 %
18	20.28	0.731	3.60 %
19	21.55	0.234	1.09 %
20	20.42	0.594	2.91 %

The twelve points marked by crosses in Fig. 5 were used to do pose estimation [6] for each frame. Table 4 shows the errors in computing the 3D locations of the remaining 20 points (marked by circles and numbered in Fig. 5). The point numbers in Table 4 correspond to numbered circled points in Fig. 5. The depth of each point from the first camera coordinate frame is also shown⁸. The average error for the twenty points used was 0.27 feet. The maximum error was 0.731 feet and the minimum error was 0.019 feet. The average percentage error was 1.22 %. The reader must note that this average is just over a set of 20 points. There are points in the sequence for which the error is much larger than 1.2 %. Points 1-4 in Table 4 have large errors because they were not localized accurately. The line-finding algorithm was not able to correctly find the borders of the lights. Points 18 and 20 have large errors because they are close to the point where the rotation axis pierces the image plane. These points therefore do not have large disparities. Points 17 and 19, which are a little further away, have correspondingly smaller errors. Finally, as noted above the imaging system has not been calibrated. Since we used a higher field of view lens for this experiment (40 deg. as compared to 24 deg. for the BOX sequence), the 3D results are more sensitive to errors in locating the image center.

⁸Since the plane of motion was roughly parallel to the image plane, these depths are approximately constant for the entire sequence.

5 Inaccurate Estimates of the Focal Length

The focal length of the lens supplied by lens manufacturers are generally quite accurate. However, when the lens is focussed on points close to the camera (i.e. when the camera is not focussed to infinity) the *effective* focal length of the system must be established by a calibration procedure [10]. In this section, the effects of incorrect estimates of the focal length on the output of the pose refinement process is examined.

The image projection (x, y) of a world point X_w given an estimate of translation T_c and rotation R_c is:

$$x = f \frac{(R_c(X_w) + T_c)_x}{(R_c(X_w) + T_c)_z} \quad (36)$$

$$y = f \frac{(R_c(X_w) + T_c)_y}{(R_c(X_w) + T_c)_z} \quad (37)$$

Dividing these two equations, we obtain:

$$\frac{x}{y} = \frac{(R_c(X_w) + T_c)_x}{(R_c(X_w) + T_c)_y} \quad (38)$$

The rotation operator can be represented as a (3x3) matrix:

$$R = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} \quad (39)$$

where s_i , $i = 1, 2, 3$ are the vectors corresponding to the rows of the rotation matrix R_c . Substituting (39) into (38), equation (38) can be rewritten as:

$$\frac{x}{y} = \frac{(s_1 \cdot X_w + T_{cx})}{(s_2 \cdot X_w + T_{cy})} \quad (40)$$

This is a linear equation in the pose parameters s_1, s_2, T_{cx} and T_{cy} which can be rewritten as:

$$x(s_2 \cdot X_w + T_{cy}) - y(s_1 \cdot X_w + T_{cx}) = 0.0 \quad (41)$$

One such equation is obtained for each world/ image point correspondence. Given 5 or more point correspondences, we can therefore solve the above system of equations and get estimates of the parameters s_1, s_2, T_{cx} and T_{cy} . The rotation parameters s_1, s_2 , however, have quadratic constraints and therefore the above system of equations must be solved by non-linear techniques. Tsai [10] uses the same system of equations in his camera calibration algorithm. Since the rotation matrix is an orthonormal matrix, estimates of its first two rows s_1 and s_2 can be used to obtain the third row s_3 using the symmetric and other orthonormal properties of the matrix. The only pose refinement parameter not determined by the system of equations is therefore the translation along the optical axis T_z .

Our goal in this section is to examine the effect of incorrect estimates of the focal length on the output of a pose refinement algorithm. Equations (40) and (41) do not depend on the focal length and consequently an incorrect estimate of the focal length would not affect the solution of these equations. Therefore, an incorrect estimate of focal length should affect only the T_z parameter of the pose; all other parameters should not change

since their estimation does not depend on knowledge of the focal length.

In practice [6] we may minimize other error functions to do pose refinement. Based on the above analysis we hypothesize that an incorrect estimate of the focal length would only significantly affect the T_z component of the pose parameters for other pose refinement methods⁹. This hypothesis has been supported by experiments using both synthetic and real data and the pose refinement algorithms described in [6]. Results of some of these experiments are shown in Table 5.

The experiments were performed using the synthetic, hallway and box image data sets described earlier. In each case, we ran the pose refinement algorithm using the correct focal length and incorrect estimates of the focal length. The incorrect estimates of the focal length were obtained by multiplying the correct focal length by a scale. Thus, in Table 5, entries in rows with focal length scale 1.0 correspond to experiments with the correct focal length and entries with rows corresponding to scale not equal to 1.0 correspond to experiments with incorrect focal lengths. Both the translation and rotation results of the pose are shown in Table 5. The rotation is shown by its angle-axis representation. The axis vector is a unit vector. As can be seen from Table 5 the only large change in any of the pose parameters for any of the experiments is in the T_z component of the translation.

Although poses can be obtained whose projection fits the original image data fairly well in the case of incorrect estimates of the image center, this is not the case for incorrect estimates of focal length. Changing the focal length causes the projection of 3D points to be dilated or contracted by a constant amount while changing the T_z component of the translation causes the image projections to dilate or contract based on their depth from the camera. As we have seen, however, the minimum of the pose error functions given incorrect estimates of focal length, leads only to a significant change in T_z . This property of poor fits makes it comparatively easier to calibrate imaging systems for focal length as compared to calibrations for the image center.

Acknowledgements

Ross Beveridge and Harpreet Singh Sawhney provided some of the input data. Raghavan Manmatha and Renee Kumar edited drafts of this paper and helped prepare the figures.

References

- [1] G. Adiv, "Interpreting Optical Flow," *PhD thesis, COINS Tech. Report 85-35*, Univ. Of Mass. at Amherst, MA., 1985.
- [2] G. Adiv and E. Riseman, "Recovery of 3-D Motion and Structure from Image Correspondences using a Directional Confidence Measure," *COINS Tech.*

⁹That is, methods where the pose is not estimated by solving the system of equations defined by (41) but by some other system of equations.

Table 5: Rotation and translation as computed by the pose refinement algorithm for the same sets of images with different focal lengths.

FOCAL LENGTH SCALE	TRANSLATION			ROTATION			
				ANGLE	AXIS		
	T_x	T_y	T_z	deg.	A_x	A_y	A_z
SYNTHETIC DATA							
1.000	4.004	-3.994	60.011	120.015	-0.577	0.577	0.577
0.928	4.013	-4.002	58.048	120.016	-0.577	0.577	0.578
HALLWAY IMAGE							
1.000	4.055	-3.942	39.926	119.808	-0.576	0.574	0.582
0.928	4.023	-3.962	37.382	119.344	-0.579	0.574	0.580
1.045	4.072	-3.932	41.509	120.066	-0.575	0.574	0.583
BOX IMAGE							
1.000	-8.256	74.647	620.313	132.833	-0.178	0.952	-0.250
1.100	-8.344	74.801	684.354	132.627	-0.177	0.952	-0.249

Report 88-105, Univ. Of Mass. at Amherst, MA., 1988.

- [3] J. Ross Beveridge, R. Weiss and E. Riseman, "Optimization of 2-Dimensional Model Matching," To be presented at the IEEE International Conference on Pattern Recognition, 1990.
- [4] O.D. Faugeras and G.Toscani, "Camera Calibration for 3D Computer Vision", *Proceedings International Workshop on Machine Vision and Machine Intelligence*, Tokyo, Japan, Feb 2-5,1987.
- [5] B. K. P. Horn, "Relative Orientation," *International Journal of Computer Vision*, Vol. 4, pp. 59-78, 1990.
- [6] R. Kumar and A.R. Hanson, "Robust Estimation of Camera Location and Orientation from Noisy Data with Outliers," *Proc. IEEE Workshop on Interpretation of 3D scenes*, Austin, Texas, Nov. 1989.
- [7] R. Manmatha, R.Dutta, E.M. Riseman and M. Snyder, "Issues in Extracting Motion Parameters and Depth from Approximate Translational Motion", *IEEE Workshop on Visual Motion - Proceedings*, March 1989, pgs 264-272.
- [8] R.K. Lenz and R.Y.Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10 # 5, pp. 713-719, 1988.
- [9] H. Sawhney and J. Oliensis, "Image Description and 3D Interpretation from Image Trajectories under Rotational Motion," *COINS Tech. Report 89-90*, Univ. Of Mass. at Amherst, MA., 1989.
- [10] R. Y. Tsai, "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision," *IEEE Int. Conf. Computer Vision and Pattern Recognition*, pp. 364-374, 1986.
- [11] L. R. Williams and A. R. Hanson, "Translating Optical Flow into Token Matches and Depth from Looming", *Second Int. Conf. on Computer Vision*, pp. 441-448, 1989.

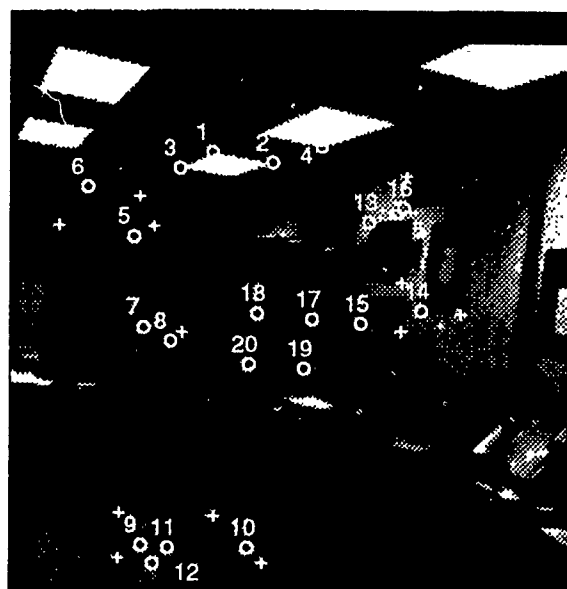


Fig. 5: Puma Image. Points marked by crosses used for pose. Numbered Points marked by circles used for depth estimation.

Recovering 3D Information from Complex Aerial Imagery

Yuan C. Hsieh, Frederic Perlant, David M. McKeown

Digital Mapping Laboratory
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213¹

Abstract

The goal of automated cartography is to generate an accurate three-dimensional model of man-made objects and natural terrain. Some of the most challenging problems in cartography exist in dense urban areas where the level-of-detail and scene clutter greatly complicate traditional feature extraction techniques. In this paper, we describe a comprehensive stereo analysis system developed to recover the three-dimensional description of an urban area using high-resolution aerial imagery. Given an area of interest in terms of geographic coverage, our system can automatically find the appropriate stereo pair using a spatial database, select control points to register the two images so that epipolar geometry is satisfied, and recover disparity information using two complementary matching techniques.

We do not assume that the initial input images satisfy the epipolar geometry constraint as this is rarely the case in unrectified aerial imagery. Therefore, we believe that stereo mapping research must *explicitly* address error and uncertainty in both scene registration and stereo matching. We show how a robust registration can be achieved using five different image domain features that are automatically extracted and selected as control points for fine image registration. In the stereo matching process two techniques are utilized, an area-based and feature-based stereo matcher, to generate a disparity map for a scene. We describe in some detail a new algorithm for stereo matching based upon one-dimensional waveform matching. We show the results of each matcher on several complex scenes and the results of a merging process that attempts to fuse these disparity maps. Finally, we describe techniques to generate a rigorous performance analysis to compare stereo matching algorithms based upon a manually derived three-dimensional ground-truth segmentation.

1. Introduction

The traditional method for obtaining a three-dimensional model of the terrain and man-made structure involves matching of stereo-pair imagery. Algorithms for stereo correspondence can be grouped into two major categories: area-based and feature-based matching¹. Both classes of techniques have advantages and disadvantages that depend on the task domain and the three-dimensional accuracy required. Area-based approaches tend to be more robust in scenes containing a mix of buildings and open terrain. However, for complex urban scenes, feature-based techniques appear to provide more accurate information in terms of locating depth discontinuities and in estimating height. No single technique performs well in both circumstances. It is precisely for this reason that we are investigating both methods of stereo matching with a goal of utilizing multiple results to achieve more accurate and robust three-dimensional interpretations.

In both area-based and feature-based techniques, the epipolar constraint is used to simplify stereo matching by reducing it to a one-dimensional problem. This is usually achieved by registering the stereo imagery. The assumption that the scene registration is ideal and that the epipolar constraint is totally satisfied, however, is rarely warranted in imagery digitized from aerial photography. Careful local registration is often required after the scenes have been coarsely aligned. Local registration needs a set of control points that are abundant, well distributed throughout the scene, and can be matched in the stereo-pair. Typically, features such as road intersections have been proposed for urban areas. However, our experience indicates that no single man-made or natural feature can satisfy all of these criteria across a variety of complex urban scenes.

In Section 2 we briefly present some recent results on automatic control point detection and matching. In Section 3 we describe our two stereo matching techniques (S1,S2), with emphasis on a new feature-based matching technique (S2) that appears to give good results in complex urban scenes. We also describe our initial results in combining disparity maps derived from both stereo matchers and we show some results on a variety of scenes. Finally, in Section 4 we present a method for evaluating the result with respect to a manual three-dimensional ground-truth segmentation and we show some of our experimental results.

¹This research was primarily sponsored by the U.S. Army Engineer Topographic Laboratories under Contract DACA72-87-C-0001 and partially supported by the Defense Advanced Research Projects Agency, DoD, through DARPA order 4976, and monitored by the Air Force Avionics Laboratory Under Contract F33615-87-C-1499. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Engineering Topographic Laboratories, or the Defense Advanced Research Projects Agency, or of the United States Government.

2. Scene registration

The primary goal of stereo photogrammetry is to determine the three-dimensional position of any object point that is located in the overlap area of two images taken from two different camera positions. The determination of the orientation of each camera at the moment of exposure and the relationship between the cameras is a necessary step in the photogrammetric process. The camera orientation determines the relationship between the image points and ground points in the scene. The classical epipolar geometry for stereo imagery establishes a very simple spatial relationship between corresponding points in the left and right images. The solution to the general camera orientation problem has four components: the interior orientation, the exterior orientation, the relative orientation, and the absolute orientation.

The epipolar geometry constraint causes conjugate points to lie on the same scanline in the left and right image. This means that stereo matching can be restricted to a one dimensional search along the common scanline. Knowledge of the maximum disparity in the scene, and ordering of matches can be used to further restrict search along the epipolar line. This constraint is used as a common framework for most stereo matching algorithms:^{2,3,4,5}. However, these stereo matching techniques assume that the registration is ideal and that the epipolar constraint is completely satisfied. For many applications in aerial image analysis one is often simply given overlapping images or partial image areas where the epipolar geometry must be derived.

In the following section we present two methods for scene registration given overlapping stereo imagery. The first method performs a coarse registration using landmarks from a spatial database. The second method uses pairs of

corresponding points in the two images to perform a relative orientation. As we will see, many of the techniques used in computer vision to establish scene registration are approximations to the photogrammetric ideal. These approximations cause the scene registration to be inaccurate and have to be taken into account by the matching process.

2.1. Coarse registration using a spatial database

The most common method to establish the relative orientation between two images is to select pairs of corresponding points in the two images. One alternative method is to independently tie each image to a common frame of reference. A cartographic coordinate system such as <latitude,longitude,elevation> is one possible frame of reference. Thus, the two images are related to a ground coordinate system, or map. The use of landmarks with known <latitude,longitude,elevation> is a common method to orient each image. The overall accuracy of the registration is dependent on the accuracy of the three-dimensional position of the landmark and the accuracy with which we can recover the image position of the landmark. We use the landmark database component of CONCEPTMAP, a spatial database system that integrates imagery, terrain, and map data to provide landmark descriptions^{6,7}. Typically CONCEPTMAP provides a registration accuracy of between ten to thirty meters for imagery digitized to a 1.3 meter ground sample distance.

Figure 2-1 and 2-2 show a stereo image pair of an industrial area taken from the CONCEPTMAP database. These images were digitized from standard nine inch format mapping photography taken at the altitude of 2000 meters using a camera with a 153 millimeter lens. One pixel in the image approximately corresponds to 1.3 meters on the ground. The left image is a 512 x 512 sub-area selected from a 2300 x 2300



Figure 2-1: Left image DC38008 with CONCEPTMAP database registration

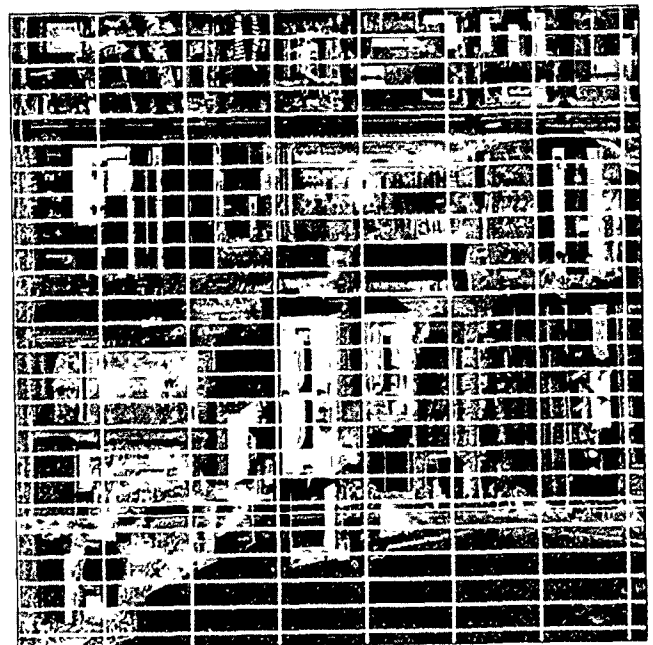


Figure 2-2: Right image DC38007 with CONCEPTMAP database registration

image. The right image sub-area was generated by calculating the <latitude,longitude> for the corner points of the left image and projecting those points onto the complete right image. This projection is then used to extract the image sub-area from the complete right image. We have superimposed a set of gridlines on both images in order to make it easier to see the actual misregistration.

2.2. Fine registration using image control points

As we have seen, the computation of the relative orientation can be accomplished by selecting pairs of corresponding points in the two images. After the relative orientation is calculated, the two images can be transformed so that they satisfy the epipolar constraint. We begin the fine registration with the coarse registration described in the previous section. We assume the transformation between the left and right image is isometric, (i.e. only translation and rotation). After the transformation, the epipolar lines correspond to the scanlines. However, problems with the accuracy of point selection led us to develop a polynomial transformation adjusted by least squares to fit the selected corresponding points.

2.2.1. Automatic selection using different features

Clearly, one requirement for automated registration is the automatic selection of corresponding points in the stereo pair images. There are two problems that must be solved. First we must automatically detect potential landmarks in each image, and then we must determine those landmarks that have been found in both images. General landmark matching is an unsolved problem and most automatic registration techniques rely on the matching of characteristic points⁸ that often have no physical significance or relationship with the landmarks.

There are some important criteria for automated control point selection. First, since the elevation of the control points is not known and we are using a simple geometrical model, it is important that the set of selected control points lie approximately in the same elevation plane. Second, the selection of control points should not rely on a single type of scene domain feature, such as road intersections, since not all control point features are abundant in all scenes. For example, in urban scenes there are often many buildings and shadow regions available as candidate control points, and they are usually well distributed throughout the imagery. However, in airport scenes elongated line pairs and uniform intensity regions appear to be a better choice. In any case we use an iterative selection algorithm⁹ that converges to a consistent set of control points that are usually a small subset of all of the possible matches in the stereo pair.

Another advantage of using multiple features for control point estimation is that the results of feature matching can be used to estimate the disparity range of the scene. Once the scene is registered, all matched features can be remapped to the new coordinate frame. It is then possible to calculate the disparity of each feature. Since all features are not at the same height, we automatically obtain a rough estimate of the

disparity range for this scene. This disparity range estimate is directly used by the stereo matching algorithms to control search for corresponding points and can greatly reduce initial matching errors. In most research stereo systems the disparity range is either manually provided or it is set to what is considered to be a "sufficiently large" value. The drawback of the former approach is that it introduces a difficult manual step in that the entire stereo model must be searched to find the minimum and maximum disparity points. The latter situation can influence the accuracy of the resulting stereo matching algorithm by causing some matches to be never considered, or decrease the efficiency by allowing large areas to be searched for which correct matches are impossible.

For this experiment, we assume that a coarse registration of the two images, such as described in Section 2.1 has already been performed. Using this coarse correspondence, we are able to limit the search to find corresponding features in the images. Most of the remaining error is translational rather than rotational which simplifies the determination of corresponding points. Candidates for automatic control point generation include shadow corners, shadow regions, BABE¹⁰ monocular building hypotheses, uniform intensity regions, and elongated line structure pairs:

Shadow corners: Shadow corners are good candidates for automatic detection and correspondence as well as for manual selection. We use corners produced by the BABE system. After removing corners that are inconsistent with shape and orientation constraints imposed by the sun direction angle and estimated shadow intensity, we select sets of shadow corners in both the left and right images. Figure 2-3 shows the corners found in the left image in white. The right image corners are shown in black and are projected onto the left image using the coarse registration. Those pairs of shadow corners that are matched are shown as connected by a white line whose endpoint circles indicate the conjugate points provided to the registration process.

Building hypotheses: Control points can also be defined geometrically with respect to features or structures extracted from the imagery. Building hypotheses generated by a monocular analysis system such as BABE can be used as match features. The center of mass of these structures is defined as the corresponding control points. Compared to shadow corners, control points defined by hypothesized buildings are not always accurate, but disambiguation of buildings is easier. Properties such as shape, size, and perimeter are good criteria that are not available for point features such as shadow corners. Figure 2-4 shows the BABE boxes in the left and right images with the matched features marked in the same manner as Figure 2-3.

Other scene features: We performed experiments to obtain control points from shadow regions, edges, and segmented regions using simple histogram analysis. In each case, control points are defined as the center of mass of the structures. Shadow regions are extracted with traditional

connected component extraction techniques, using an estimate of shadow intensity provided by BABE¹¹. Due to variation in the shape of the shadows, shadow regions usually give poor results in complex urban scenes with very high buildings. This variation of shape is caused by occlusion of the shadow by tall structures. They can be very reliable, however, in suburban house images where buildings are separated and have simple roof profiles. Edges are another feature extracted by BABE. Only edges with significant length are used as candidates for matching. The criteria for edge matching are edge orientation, length and the intensity gradient across the edge. Figure 2-5 shows the significant lines extracted and matched in the industrial scene. Finally, unique bright points in the scene can be used to form bright blob regions. The intensity threshold

for blob regions is determined by successively decreasing the intensity scale until enough regions are extracted. These features turned out to be useful for scenes with few or no man-made structures, where shadow corners, hypothesized buildings and shadow regions failed to generate enough matching candidates.

Figure 2-6 shows the superposition of BABE results using the refined registration from Figure 2-4. The offset between building hypotheses is now primarily in the column direction and can be attributed to the displacement of the building in the left and right image due to their height. In many cases we have been able to automatically reduce the row offset error to sub-pixel accuracy from an initial displacement of 15 to 20 rows in the coarse CONCEPTMAP registration.



Figure 2-3: Shadow corners selected



Figure 2-4: BABE building hypotheses selected

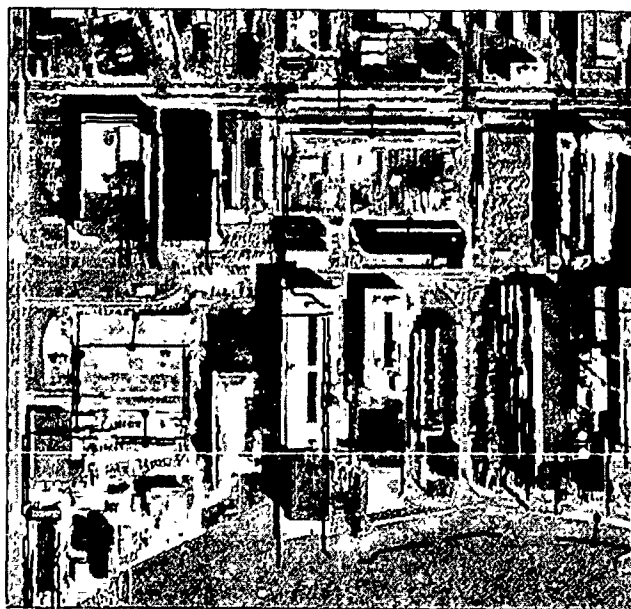


Figure 2-5: Significant lines selected

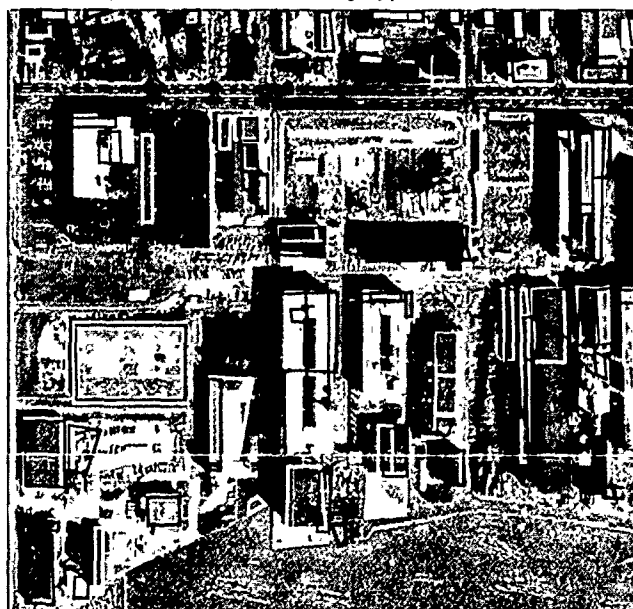


Figure 2-6: Fine registration using BABE points

2.2.2. Evaluation of automatic registration

Table 2-1 shows the local accuracy of the different scene registrations performed on the industrial scene shown in Figures 2-1 and 2-2. POLY means that actual registration is performed using a polynomial fit, whereas ISO means that the images are registered using an isometric solution. Coarse registration is the result of CONCEPTMAP registration. Using a set of manually selected control points we are able to evaluate the accuracy of each registration in terms of row offset compared to the ideal epipolar geometry (corresponding points on the same scanlines). Polynomial approximation performs better overall than isometric approximation, but it is more sensitive to noise. Further, the isometric approximation only requires three control points. For this scene, there are enough points from any of the match features to compute a second order polynomial approximation. The resulting accuracy is comparable with that achieved using manual selection of control points.

In summary, scene registration is a key initial step in many tasks involving the automated interpretation of aerial images. Stereo analysis requires particular care in scene registration because of the geometric assumptions made by most stereo matching algorithms and their inability to recognize and recover from registration errors. Such registration errors usually end up reflected as gross errors in the stereo match. As a part of our goal to produce three-dimensional interpretations of complex urban scenes we have found it necessary to develop registration techniques that are accurate and robust across a variety of scene domains. We have tested our system on airport scenes, urban scenes, and suburban housing developments with varying degrees of success. We are currently investigating ways to evaluate the distribution of control points and to incorporate this evaluation into the registration system. We are also looking into improving our

ability to recover more feature-based control point descriptions based upon other feature extraction systems, such as road detection and tracking¹².

3. Two Stereo Matching Techniques

Algorithms for stereo correspondence can be grouped into two major categories: area-based and feature-based matching. Area-based techniques may provide a dense disparity map with an estimate generated at every point in the image. Feature-based approaches provide depth information only at points where the features are generated, often points of intensity discontinuity that may correspond to discontinuities in depth.

We do not believe that any one technique is likely to be robust enough to perform well under the diverse set of conditions found in urban areas. For complex urban scenes, feature-based techniques appear to provide more accurate information in terms of locating depth discontinuities and in estimating height. However, area-based approaches tend to be more robust in scenes containing a mix of buildings and open terrain. For this reason we have developed two stereo matching algorithms. S1 is an area-based algorithm and uses the method of differences matching technique developed by Lucas^{13,14}. S2 is feature-based using a scanline matching method that treats each epipolar scanline as an intensity waveform. The technique matches peaks and troughs in the left and right waveform. Both are hierarchical and use a coarse-to-fine matching approach. Each is quite general as the only constraint imposed is the order constraint for the feature-based approach. The order constraint should generally be satisfied in our aerial imagery except in the case of hollowed structures.

Both matching algorithms assume the imagery has been registered into the epipolar geometry as discussed in the

Statistics on the quality of different registration for DC38008						
Type of Registration	Number of points	Avg. row offset	Std. row offset	Min/Max row off.	Avg. col offset	Std. col offset
Coarse	-	-20.4	1.6	-23/-16	0.4	1.2
POLY manual	11	0.1	0.3	-1/1	0.1	0.5
POLY corner	20	0.5	0.6	0/2	-0.5	1.2
POLY structure	14	-0.8	0.8	-2/2	-5.2	2.0
POLY edge	17	0.8	0.7	0/3	0.0	1.8
POLY shadow	12	-0.6	0.8	-2/1	-0.4	0.9
POLY blob	17	0.6	0.6	0/2	-0.6	1.1
ISO manual	11	-0.4	0.6	-1/1	0.6	1.4
ISO corner	20	1.0	0.5	0/3	2.7	1.3
ISO structure	14	-1.7	0.9	-3/1	-2.9	1.2
ISO edge	17	1.3	0.9	0/4	0.9	1.2
ISO shadow	12	-0.2	1.1	-2/2	3.8	1.7
ISO blob	17	0.6	1.6	-2/5	1.4	1.2

Table 2-1: Statistics for different registrations on DC38008 stereo pair

previous section, and each algorithm produces a disparity map that is registered to the left stereo pair image. Both matching algorithms also need an estimate of disparity range found in the image. This estimate need not be perfect, but the accuracy of the disparity range estimate directly affects the quality of stereo matching. An accurate disparity estimate limits the search range and therefore, reduces possible mismatches. The disparity range can be provided by a user or from the result of registration process. To generate disparity ranges automatically, we first need to select a type of registration to use, and register all features described in 2.2.1 using the same type of registration. Once the control points from both left and right images are in the same coordinate frames, the column offset between a pair of matched control points is the disparity. We can find a good approximation to the scene disparity range by examining the predicted disparity value for each of the control points⁹. Table 3-1 shows the true disparity range, the disparity range selected automatically, and the manually selected disparity range. The disparity range used for all matching examples shown in this paper were selected manually. This was primarily to simplify the stereo performance analysis and avoid confusing errors in the disparity estimate with matching errors generated by the two stereo systems. In most cases there is little difference in the match quality using either automatic or manual disparity estimates. As we have stated earlier, we believe that researchers need to address the problem of accurate automatic disparity range estimation as a component of a fully automated stereo analysis system. The alternative, manual analysis, is both tedious and prone to error especially in complex scenes containing large elevation jumps due to man-made objects and natural terrain.

Disparity Range Statistics (in unit disparity)			
Scene	Ground Truth	Automatic	Manual
DC38008	-2:14	-4:17	-5:15
DC37405	-13:13	-14:12	-15:15

Table 3-1: Statistics of disparity range for DC38008 and DC37405

3.1. S1: Method of differences

The S1 area-based approach uses a hierarchical set of reduced spatial frequency images to perform coarse-to-fine matching on small windows in the two images. At each level, the size of the windows for the matching process depends on the spatial frequency resolution (smoothness) of the image. An initial disparity map is generated at the first level. Subsequent matching results, computed at successively finer levels of detail, are used to refine the disparity estimate at each level. Therefore, the amount of error in the scene registration that can be tolerated by this matching algorithm depends on the size of the matching windows. However, since there is a relationship between the matching window size and the level of accuracy, simply using larger matching windows may not be desirable. Consider a point in the left image of the stereo pair; the difference between the correct disparity value and our initial

estimate is the amount by which the stereo process must correct the disparity. Initially this difference will be relatively large because the initial disparity estimate is not particularly accurate. Because of this, the method of differences requires that we start out with smoothed images to accommodate these large differences. As the disparity estimate improves, we can use less smoothed images because the magnitude of the matching error decreases.

S1 does not require sensitive feature extraction thresholds as is common for feature-based approaches. Stereo matching in S1 is accomplished for every pixel and is not restricted to selected image features such as interesting areas, edges, lines, or other extracted features. S1 is not overly reliant on perfectly registered stereo pairs taken simultaneously by well-parameterized cameras, nor does it require threshold tweaking to accommodate matching of edges or vertices. In the resulting S1 disparity images the speckled areas are caused by loss of correspondence in large nondescript sections. Such nondescript sections are characterized by the lack of edges or texture. The boundaries of the objects are fuzzy and the big discontinuities are not well captured. S1 performs well with objects having height but often, when it is initiated with too small a disparity range, it will not converge to a correct result. Thus, an approximate estimate of the expected disparity range is required.

To accommodate large disparities, we use a hierarchy of different spatial resolutions. Starting with a reduced spatial resolution data set we compute an initial estimate of the scene disparity. With this estimate of disparity as an initial starting point, we can better refine our estimate than if we had begun matching at a coarser level. The disparity range of the scene can be used to estimate the number of different spatial resolutions, the number of levels for each resolution, and the size of the smoothing windows and scanning overlap at each level. A good estimate of the disparity range can be provided by shadow analysis, matched features, or external knowledge of the terrain. We have found that good estimates of the disparity range are necessary to achieve reasonable results. This approach has been used on different images and gives better results than the standard S1 method. The results are less sensitive to registration errors and we obtain better results on the discontinuities.

As a final step in the S1 analysis we attempt to improve the detection of the disparity discontinuities. We first compute variational left and right images using a local variation operator¹⁵. As an initial disparity estimate, we then use the result of the previous method and rerun the S1 procedure using just two resolution levels with the variational images to encompass errors in the previous result, and thereby locally refine the disparity estimate.

3.2. S2: A Feature Based Approach

S2 is a feature-based system that treats the problem of stereo matching as one-dimensional signal matching, similar to Witkin's scale space signal matching¹⁶. S2 extracts intensity and gradient signals from each epipolar scanline and matches peaks and valleys in the left and right signal. S2 is hierarchical and uses a coarse-to-fine matching approach to guide successive refinements of the waveform approximation. This work takes a similar approach to previous work in stereo matching. Baker¹⁷ used a coarse-to-fine approach with a Viterbi dynamic programming algorithm to match edges. Ohta's⁵ technique tried to find an optimal matching surface in a three-dimensional search space using inter-scanline search and dynamic programming. However, edge matching is difficult, especially in complex scenes, often requiring thresholds based upon edge strength to avoid combinatorial explosion. Edge matching methods also rely heavily on the quality of edge detectors and the resulting disparity maps are usually very sparse. Our approach is to view intensity profiles of scanlines as waveforms. There has also been some previous work using this paradigm in the pattern analysis area to perform waveform correlation by representing the waveforms as trees¹⁸, or by tracking zero crossing of intensity signals through scale space¹⁶.

S2 matches epipolar scanlines in the left and right image using a hierarchical approximation of the scanline intensity waveform. It matches peaks and valleys in the waveform at different levels of resolution. S2 uses intra-scanline consistency to enforce a linear ordering of matches without order reversals. It also applies an inter-scanline consistency that considers the matches in adjacent scanlines. Application of the inter-scanline constraint is used to increase the confidence of matches found to be consistent across multiple scanlines and to delete improbable matches. Since disparity discontinuity usually occurs at the intensity discontinuity, the gradient waveform is matched after the intensity matching phase to localize disparity jumps. Finally, efforts are made to detect occlusions and correct them. An overview of the matching procedure in S2 is as follows:

1. Coarse-Level approximation of intensity profile. Match significant features found at this level.
2. Mid-Level approximation of intensity profile. Match significant features found at this level.
3. No approximation. Match all possible features.
4. Intra-scanline consistency check.
5. Inter-scanline consistency check.
6. Gradient Matching.
7. Post Processing.

3.2.1. S2 Hierarchical Waveform Matching

The first step in the S2 matching algorithm is to obtain waveforms from the left and right stereo pair images. This is accomplished by extracting an intensity profile from the epipolar scanlines in each image. Peaks and valleys in the intensity profile are identified and these points are used as *matchable features* for S2.

S2 uses a coarse-to-fine approach. We match a subset of the matchable features that are most significant and, therefore, it is relatively easy to find unambiguous correspondences. We then use this initial correspondence to constrain the matching of the full set of features. We use a divide-and-conquer line fitting algorithm¹⁹ to *approximate* the original intensity profile. This algorithm is used because it identifies significant features, maintains the *general* shape of the original intensity profiles and the position and magnitude of the peaks and valleys.

Features are matched according to their relative similarity. A similarity function is used to measure the *similarity* between two features. This similarity function has three components: intensity, shape, and description.

Intensity component: The intensity component is a form of 1D correlation working on a normalized intensity profile. Intensity profiles are normalized to account for the difference in contrast that may occur in stereo pair imagery.

Shape component: The shape component uses an approximated waveform to measure shape of the features. During the waveform approximation, we produce a piece-wise approximated line. This line is composed of segments whose end points identify the significant features. The shape component measures the similarity of the length and angle for corresponding triples of significant features. The shape evaluation is performed for the center feature point and is only

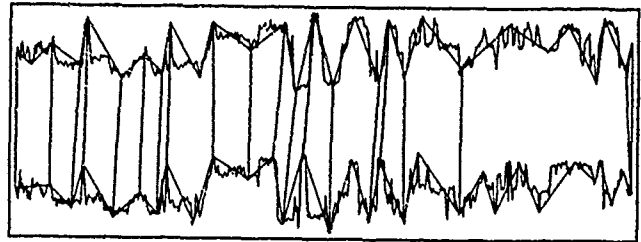


Figure 3-1: Intensity waveform matching at coarse level

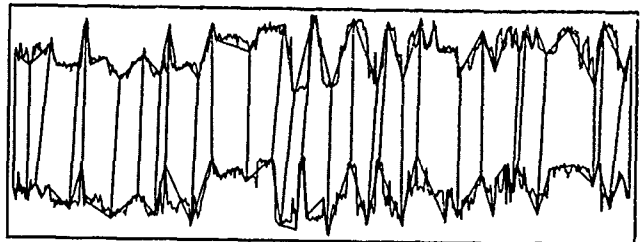


Figure 3-2: Intensity waveform matching at middle level

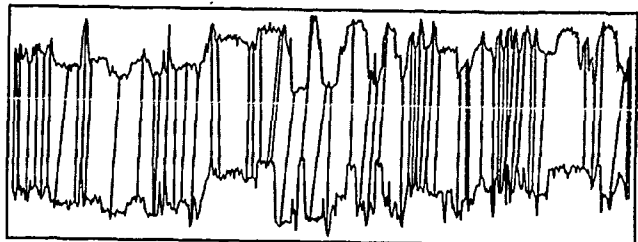


Figure 3-3: Intensity waveform matching at fine level

computed during the first two match phases since third phase does not use an approximated waveform.

Description component: The description component is used to disambiguate features with more than one possible match. It uses a significant feature labeling that disallows certain waveform match combinations, such as 'peak' with 'valley'. It also takes into account local minimum and local maximum of the significant features.

Figure 3-1 shows the left and right intensity waveform and the matching results at the coarse level of approximation. The waveform on the top is the profile from the left image, while the one on the bottom is from the right. The jagged lines are the approximation of the intensity waveform. Line segments connecting left and right waveform are the matches made. Figures 3-2 and 3-3 shows the intensity waveform matching at mid-level approximation and fine-level full resolution waveform matching.

As with any matching algorithm the local optimal matches at the waveform level might not be correct from a more global point of view. It is precisely for this reason that inter- and intra-scanline consistency constraints are imposed during the intensity matching phase^{5,17}. Inter-scanline consistency simply assumes that disparity should be nearly continuous across the scanlines. Intra-scanline assumes continuity along the scanline, unless there is strong support for the disparity jump. Strong support for a match exists if two features are similar and there exists no better match, or match with the same similarity, in the neighborhood. Both inter- and intra-scanline consistency are accomplished by looking at the neighborhood of a match and trying to establish a consensus. The intra-scanline check is performed at the end of each intensity matching phase. Inter-scanline checking is performed after all three matching phases have been completed. If a

match is not consistent with its neighboring matches then S2 checks to determine if a new match can be made with sufficient confidence. A new match has sufficient confidence if the combination of similarity score and consistency score is better than the old match. The similarity score is computed in the same way as during the last intensity matching phase and the consistency score is the standard deviation of the disparity estimates of all the matches in the neighborhood.

Gradient matching occurs after inter-scanline matching. S2 converts the intensity profile into a gradient profile. Since we are only interested in the intensity jump along the profile, only the x component is relevant. We treat the gradient profile identically to an intensity waveform and proceed to match the epipolar gradient waveforms using the fine resolution intensity matches as constraints. Since we already have a dense set of intensity matches there is no need to perform a coarse-to-fine approximation for the gradient. We assume that there are sufficient matches from the intensity phase to provide match constraints in the gradient domain.

S2 performs a final post processing step to explicitly deal with the problem of boundary occlusion. We can detect an occlusion using the gradient profile when we find unmatched significant features in one profile that occur between two successive good matches where one match is a high disparity estimate and the other is a low disparity estimate. This situation is identified and corrected by allowing a two-to-one feature match. In other words, a extra feature in one profile is matched to a feature in the other profile that already has a match. At the end of this phase, we can create a sparse disparity map as shown in Figure 3-4. Points in this image represent the actual matches found by S2 and are only a small subset of the three-dimensional point in the scene. In the following section we describe the interpolation of this sparse disparity map into a dense disparity map to recover height estimates for the entire scene.

3.2.2. Interpolation

One key issue in feature-based stereo matching is the interpolation process. Because we are obtaining depth estimates at sparse matching points, we must fill in depth estimates in a consistent manner in order to achieve a complete disparity estimate. There has been much work done in surface interpolation techniques; some combined the interpolation process into normal stereo processing^{20,21}, while others tried surface fitting with sparse data²². However, we have not found a satisfactory technique that works in both urban environments with large disparity jumps as well as in smoothly varying terrain. At present, a constant step interpolation is used because it is the most suitable method given the sharp disparity discontinuities found in urban scenes.

Figure 3-4 shows the result of the S2 process in the industrial scene. White points are actual match points while black pixels correspond to points with no disparity estimate. Figure 3-8 shows the result of interpolating the sparse disparity map



Figure 3-4: S2 sparse disparity map

smoothed by a vertical median filter. Figure 3-8 shows that S2 performs well on discontinuities with most of the mismatches and errors occurring at the occlusion boundaries. In the following section we show stereo matching results for two complex urban scenes, DC38008 and DC37405, and for a scene containing rugged terrain, AIV.

3.3: Stereo Test Results

S1 and S2 have been tested on approximately fifteen stereo scenes including airports containing hangars, runways, and tarmac, suburban house areas with complex terrain and buildings, and industrial areas with large and complicated buildings. The results presented here are representative of problems and successes achieved in these tests. We present the

reference left image, a reference disparity map that has been manually compiled using an interactive 3-D editing system, and the S1 and S2 results. In all of the disparity map results presented in this paper, brighter regions are closer to the camera and have greater height. Darker regions are at or below the relative terrain ground plane established by the scene registration process. Thus, the disparity map encodes relative height. Given several points with known absolute elevation in the scene we could calculate the absolute height at each point in the disparity map.

Figure 3-5 is a complex industrial area scene. This scene contains many of the difficulties found in stereo matching, including occlusion, complicated textures, large depth

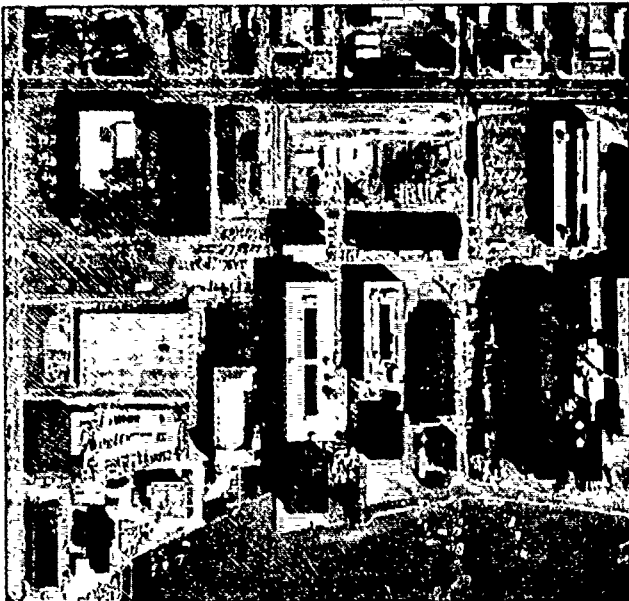


Figure 3-5: DC38008 Industrial Scene



Figure 3-7: S1 Disparity Map

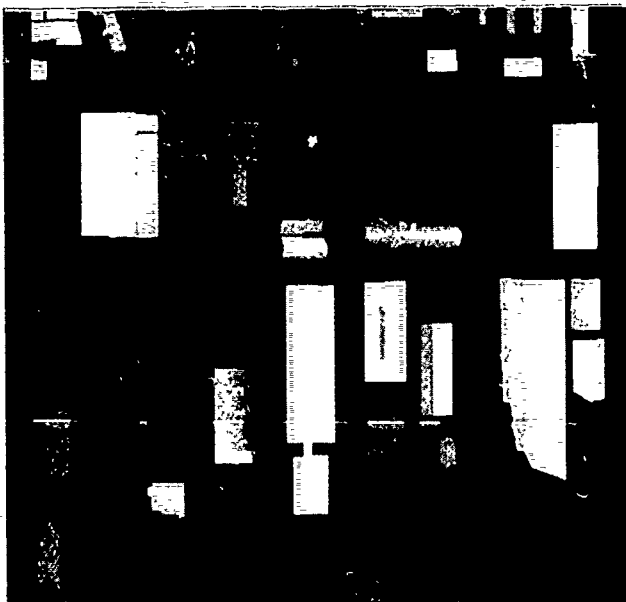


Figure 3-6: DC38008 Disparity-Reference



Figure 3-8: S2 Disparity Map

discontinuities, and complicated three-dimensional objects. Figure 3-6 shows a manually compiled disparity reference map. Figures 3-7 and 3-8 shows the result of the S1 and S2 matching respectively. S1 performs well on textured, smoothed, and continuous regions. However, the depth discontinuities are not well captured, and therefore the delineation of the high buildings is not crisp. However, most of the terrain relief is present, particularly the slope of the land toward the water in the bottom portion of the scene. One advantage of S1 matching is that it is not overly reliant on initial scene registration and can accommodate small errors without producing artifacts in the disparity map. The results of S2 stereo matching shown in Figure 3-8 are much better in terms of detection and estimation of depth discontinuities. The

overall building structures are quite apparent, including atriums and sloped roofs. However, due to the simple step interpolation some matching errors are propagated over large portions of the scene. Interestingly, this allows us to easily detect errors in the matching process that would otherwise be difficult to see had we performed a linear interpolation. In addition to pointing us to possible problems in the S2 matching algorithm, this also argues for a more detailed analysis of the disparity map with respect to other cues in the scene. Such cues include the use of shadow regions to indicate possible areas of occlusion and the analysis of intensity-based monocular segmentations to predict object surface orientations.

Figure 3-9 is a complex urban scene with hilly terrain and



Figure 3-9: DC37405 Industrial Scene



Figure 3-11: S1 Disparity Map

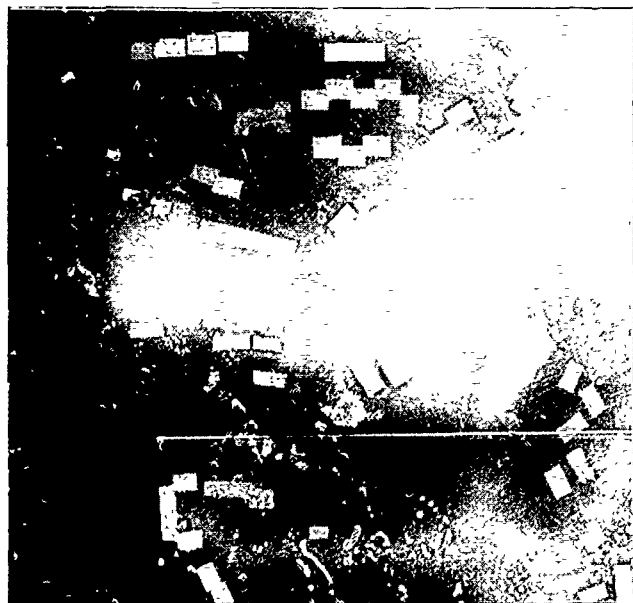


Figure 3-10: DC37405 Disparity Reference

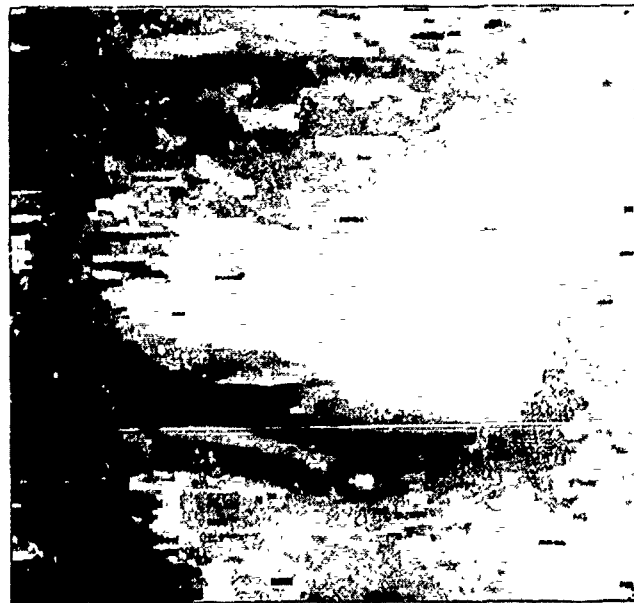


Figure 3-12: S2 Disparity Map

large number of small buildings with low height. There is also a great variety of building shapes including large apartment structures, town homes, and low density commercial buildings. Due to the complexity of the terrain there are many areas where the man-made structures have significantly lower height than the surrounding ground. Thus, it is a very good test site for detecting relative strengths and weaknesses of area-based and feature-based matchers. Figure 3-10 is the manually generated reference disparity map while Figures 3-11 and 3-12 are the results of S1 and S2, respectively. Both S1 and S2 do a

good job of recovering the complicated terrain including the road/valley. Both captured the general ground relief with S1 providing a better terrain estimate and S2 capturing the building shapes more crisply.

Figure 3-13 is a scene without any significant man-made structures. This scene was provided by the U.S. Army Engineering Topographic Laboratories, Fort Belvoir, VA., with the imagery already registered into the epipolar geometry. It



Figure 3-13: Denver ALV test site

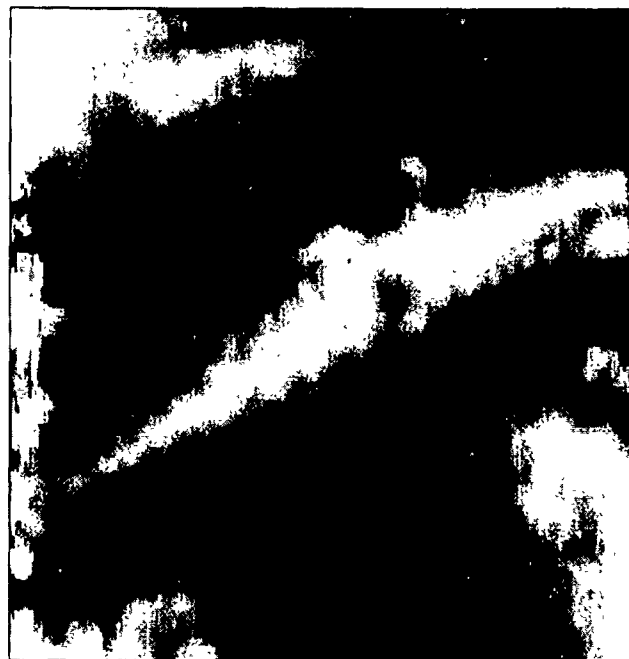


Figure 3-15: S1 disparity map for denver scene

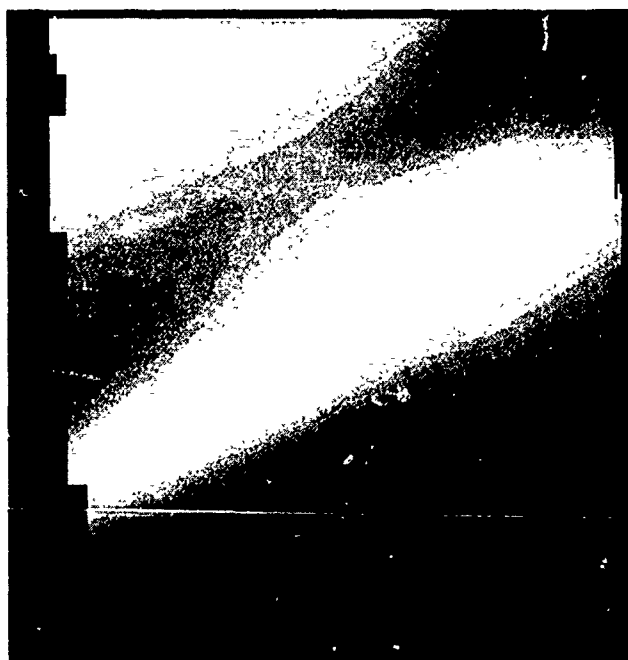


Figure 3-14: Reference disparity map for denver scene



Figure 3-16: S2 disparity map for denver scene

has a very large disparity range, compared to the previous examples, and exhibits complex terrain structure. Figure 3-14 shows manually created disparity map for this scene. Figure 3-15 and 3-16 shows the result for S1 and S2. Both methods recovered the general terrain relief in the scene. S2 had difficulties at the right hand edge of the image because that part of the scene is not visible in the right image of the stereo-pair. A more detailed analysis indicates the surprising fact that S2 is performing better (with average pixel disparity error of 9%) than S1 (with average error of 22%). This is probably due to the very large disparity range in this scene and some inherent limitations in hierarchical area-based algorithms, such as S1, when recovery of large ranges (over 25% of the scene) is required.

While improvements can be made to the intrinsic performance of each of the individual stereo matching algorithms, we believe that there is much to be gained in the fusion of disparity estimates across different stereo techniques. In the following section we describe a simple technique for merging the disparity maps generated by S1 and S2 in order to produce an improved scene disparity map.

3.4. Merging S1 and S2 Results

The disparity maps produced by area-based methods have characteristics of smooth relief and can give very accurate point disparity estimates. On other hand, feature-based methods produce disparity maps that are noisy and sparse, but have good delineation of the disparity discontinuities. Our goal is to capitalize on these different strengths to generate an improved disparity estimate. Others have recognized the utility of bringing additional information to bear in order to refine disparity estimates. For example, the Stereo Vision System¹⁵ from USC used features such as intensity edges to refine their area-based disparity map. We have taken the approach of treating the different disparity maps as different height hypotheses that can be evaluated with respect to each other at every point in the scene. The idea behind this is that at the disparity edge, the feature-based method should produce more accurate results than the area-based method, while at smoothly changing terrain, the area-based disparity value should generally be considered to be more reliable.

Presently, a very simple technique is used. The system looks at the disparity hypotheses generated by S1 and S2, and selects the better disparity value. To avoid bias, the goodness of the disparity is measured using techniques similar to the area-based and waveform based methods. An image patch is extracted from both the left and right stereo pairs. The position of the right image patch extracted depends on the disparity value. The goodness of the disparity is simply the difference in the two intensity patches. A similar measure is also used for the waveform measure.

When the disparity estimates from both methods differ by about 20% of the disparity range, we do not try to generate any hypothesis. These areas usually correspond to occluded

regions. These regions can not be matched since the corresponding regions in the other part of the stereo pair are not visible. In S1, these regions are simply filled in by matching areas with similar characteristics outside the occluded region. In S2, since there were no matchable features, the disparity values are interpolated from the nearest bounding pair of matched points.

Figure 3-17 shows the result of merging the S1 and S2 results for the DC38008 industrial scene. The black regions within the disparity map show the regions of possible occlusion. Qualitatively it appears that this result has a better delineation of disparity jumps when compared with the S1 result in Figure 3-7, and is less noisy when compared with the S2 result in Figure 3-8. Thus, the results of the two stereo matching algorithms are quite complementary. We believe that it is possible to take advantage of the different failure modalities in order to form a composite disparity map that gives a more accurate three-dimensional representation of the scene.

In this section and in Section 3.3 we have primarily invoked subjective descriptions of the relative performance for each of the matching algorithms. Unfortunately, these are precisely the type of performance descriptions found in most of the stereo matching literature. Often researchers resort to the display of a perspective view to show the three-dimensional reconstruction. Such displays or qualitative statements of performance, while not inaccurate, do not actually allow us to understand the impact of small algorithmic changes to the matching technique, the effect of various registration methods on the overall scene interpretation, or the effect of various analysis methods such as merging, in a way that is quantitative. The lack of accurate ground-truth information makes it quite difficult to evaluate stereo matching algorithms performed by various researchers even on identical imagery. In the following section we describe some quantitative evaluation methods for performance

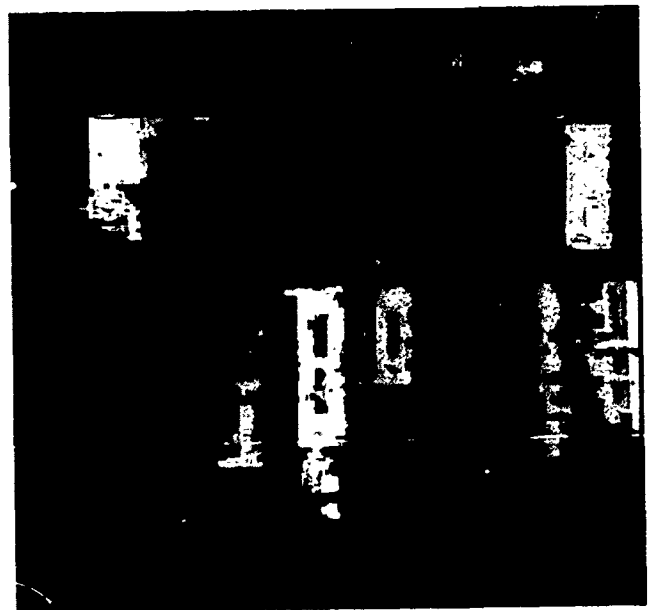


Figure 3-17: Result of Merging S1 and S2

analysis based upon the manual generation of a detailed disparity map and scene segmentation.

4. Performance Evaluation

It is difficult to quantitatively evaluate the results of any stereo matching algorithm working on real, rather than synthetic, stereo image data. While random dot stereograms can provide controlled three-dimensional scene structure we do not believe they are sufficient to evaluate stereo matching algorithms in complicated imagery with natural and man-made structures. Two different evaluations are possible. We can compare a disparity result to a reference disparity map or we can compare different disparity results to one another. A true evaluation of the results, however, requires the use of a reference 'ground-truth' disparity map for comparison.

It is actually very difficult to get a good reference disparity map for an arbitrary test scene. One could imagine resorting to the use of existing digital elevation models, or paper maps with terrain contours. Unfortunately, unless one is fortunate enough to find an area with high resolution ground-truth, the accuracy of standard digital products or maps is insufficient, especially

with a ground sample distance around 1 meter per-pixel. We have developed a display tool to manually generate disparity maps allowing a user to select points on the registered images and generate accurate disparity values. The user views the scene using a Tektronics 920 stereo display monitor with the imagery registered using a manual ground point selection. Once a sufficient number of points have been selected, usually a couple hundred, but depending on the complexity of the underlying terrain, we can generate a dense reference disparity map of the terrain by interpolation. Similarly, we add to the terrain disparity map, disparity regions that correspond to man-made structures. In some sense these manual disparity maps are detailed cartographic descriptions of the scene and can be much more accurate than most traditional paper-based maps. Figures 3-6, 3-10, and 3-14 show the manually produced disparity maps for the industrial, suburban house, and Denver terrain scenes.

At least three different performance measures can be calculated to evaluate a stereo disparity result. We can evaluate the general performance on a scene, the performance for all the buildings, or the performance on a building-by-building basis. The global average disparity error is computed

Global Error Estimate for Stereo Matching Using Figure 3-6 as ground truth				
Stereo Method	Min/Max Disparity	Average Error % (pixel disparity)	% of points within +- 1 pixel disparity	Ground Truth Disparity Range
S1	-12/13	7%(1)	58%	-2/15
S2	-5/14	6%(1)	63%	-2/15
S1+S2	-10/14	5%(1)	59%	-2/15

Table 4-1: Statistics for different stereo matching methods on DC38008

Global Error Estimate for Stereo Matching Using Figure 3-10 as ground truth				
Stereo Method	Min/Max Disparity	Average Error % (pixel disparity)	% of points within +- 1 pixel disparity	Ground Truth Disparity Range
S1	-12/12	5%(1)	63%	-13/13
S2	-15/15	4%(1)	70%	-13/13
S1+S2	-15/15	4%(1)	70%	-13/13

Table 4-2: Statistics for different stereo matching methods on DC37405

Global Error Estimate for Stereo Matching Using Figure 3-14 as ground truth				
Stereo Method	Min/Max Disparity	Average Error % (pixel disparity)	% of points within +- 1 pixel disparity	Ground Truth Disparity Range
S1	-22/19	5%(2)	61%	-28/-1
S2	-26/1	6%(1)	70%	-28/-1
S1+S2	-25/1	6%(1)	70%	-28/-1

Table 4-3: Statistics for different stereo matching methods on Denver scene

by finding the error for each point between an estimated disparity value and the reference disparity map. This single statistic provides a quick quantitative measure of the quality of the disparity map. One can further categorize points in the reference disparity map as high gradient points, low gradient points, points with high disparity, or points with low disparity. Based upon this classification it could be interesting to evaluate the performance of various stereo matching algorithms for specific problems such as smoothing over depth discontinuities or sensitivity to disparity range.

We describe statistics on the error between the reference

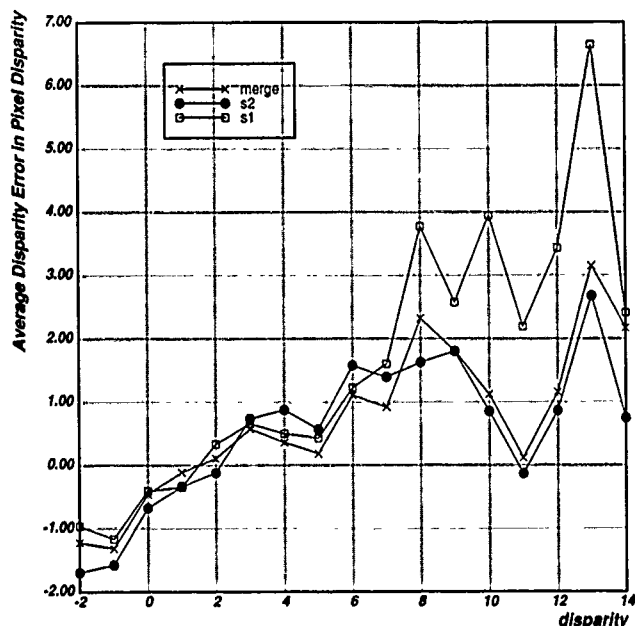


Figure 4-1: Average Error in Pixel Disparity at Each Disparity Level in DC38008

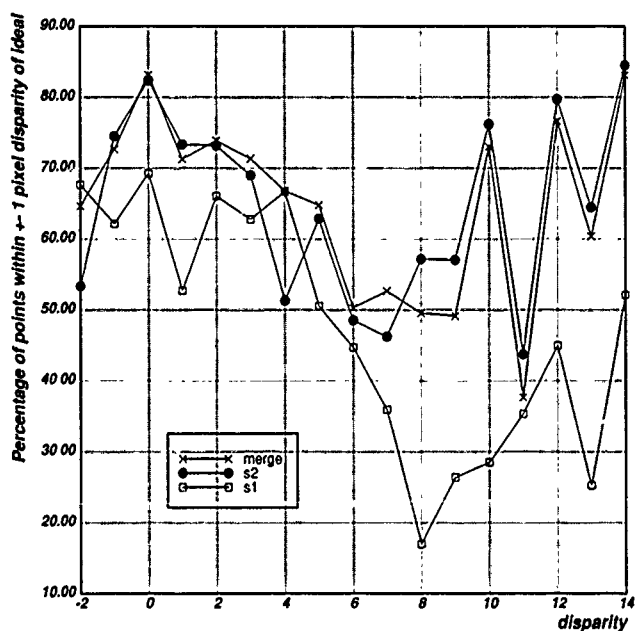


Figure 4-2: Percentage of Points within +/- 1 Pixel of Ideal Disparity in DC38008

disparity value and the disparity result without any further classification. For our global measure we present the average error for the entire scene and the percentage of points having an estimate within ± 1 pixel disparity from the reference for the entire scene. The use of ± 1 pixel disparity reflects some of the accuracy limitations in the reference disparity map and is discussed further in Section 4.3. These simple parameters give us an idea of the magnitude of the errors in the scene, but do not give much insight into their distribution. Other error metrics such as min/max error are not very reliable since they can be caused by single point errors that may occur

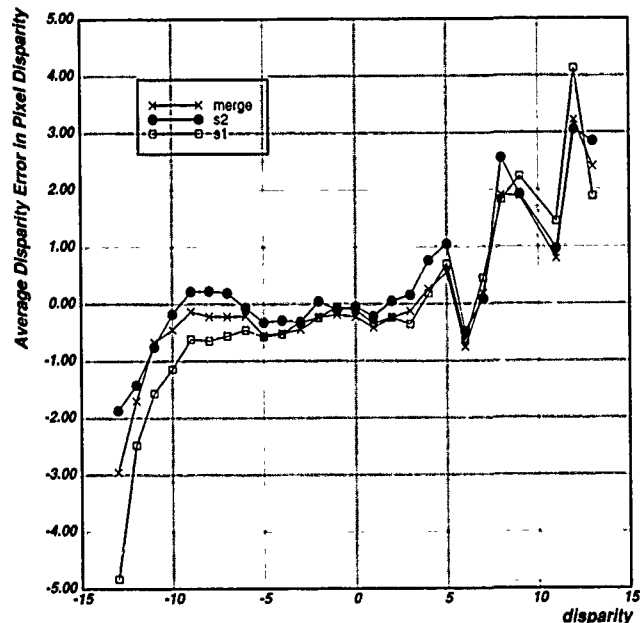


Figure 4-3: Average Error in Pixel Disparity at Each Disparity Level in DC37405

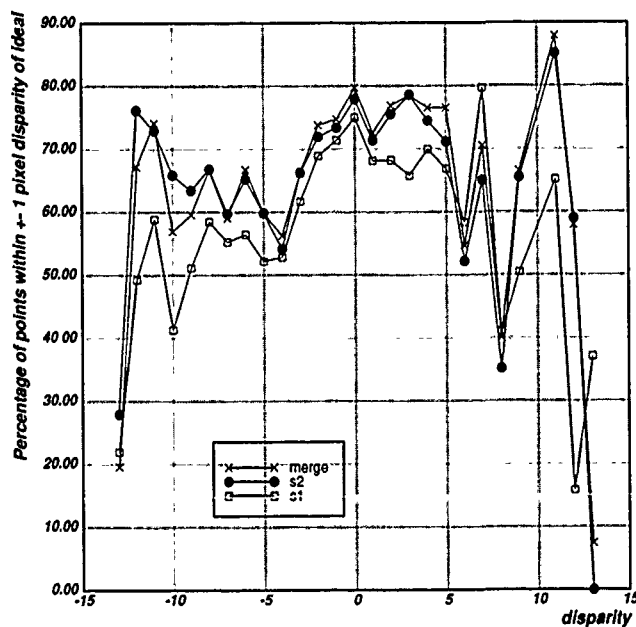


Figure 4-4: Percentage of Points within +/- 1 Pixel of Ideal Disparity in DC37405

in either the calculated or reference disparity map.

Tables 4-1, 4-2, and 4-3 give the global error estimates for each of the three test scenes. These global statistics show that S1, the area-based method, S2, the feature-band method and merge, the combination of S1 and S2, give very similar results across each of the three scenes. Interestingly, these measures do not seem to statistically reveal the apparent perceptual improvement achieved by merging the results of S1 and S2. We believe that this argues for a more structural analysis in addition to global scene measures.

One way to address some of the issues that are hidden by global statistics is to measure the influence of the disparity value on matching accuracy for each of the methods. The graphics in Figures 4-1, 4-2, 4-3, 4-4, 4-5, and 4-6 plot error rates sorted by reference disparity. Figures 4-1, 4-3, and 4-5 show the average error in pixel disparity at each disparity level for each of the test scenes. Each contains three graphs showing the results for S1, S2, and the merged result of S1 and S2. Figures 4-2, 4-4, and 4-6 show the percentage of points within ± 1 pixel of the ideal pixel disparity over each disparity range.

In general, these graphs indicate that the greater the actual disparity, the more likely the various matching algorithms will make a mistake. This is reflected in both a higher average error and a lower percentage of points within ± 1 pixel of the actual disparity. These global metrics also show that in areas of low disparity, S1, S2, and their merger give similar results. For higher disparities S1 has much more of a problem in correctly estimating the disparity than does S2. Further, in most cases, the result of S1 and S2 merging produces an improved estimate causing errors to decrease.

In areas with man-made structures global accuracy statistics do not adequately convey the quality of the stereo matching system with respect to the buildings in the scene. In most cases buildings may cover only a small portion of the scene and the background terrain will statistically dominate the scene-wide estimate of disparity quality. Thus, we require a method that allows buildings to be evaluated independently or as a class of objects in the scene. Additionally, there are several metrics that can be used to evaluate both the disparity estimate and the quality of the depth jumps. We discuss these metrics in the following sections. Figures 4-7 and 4-8 are hand segmentations of the left image where we have associated a reference building IDs. Figures 4-9 and 4-10 are graphs showing the actual building heights referenced to the building IDs. We have also computed, for each building in the ground-truth, the height of the building over its surrounding terrain. We have assigned building ID's based upon the ground-truth disparity map so that taller buildings have larger numeric ID's.

4.1. Quality of Building Disparity Estimate

In order to evaluate the performance of S1, S2 and the merged result on buildings in the scene we can gather statistics on the disparity estimate for each pixel considered to be on the roof of the building. As before, the average disparity error in pixel disparity and the percentage of points within ± 1 pixel of the ground-truth estimation are good measures for performance. Figure 4-11 shows the quality of the disparity estimate for each of the buildings in the DC38008 industrial scene. The x-axis represents the ID number for each building and the y-axis shows the errors in estimated disparity for a particular building across S1, S2, and the merged result. This graphic, although a bit cluttered, shows no clear trend of performance advantage; both S1 and S2 produce a comparable

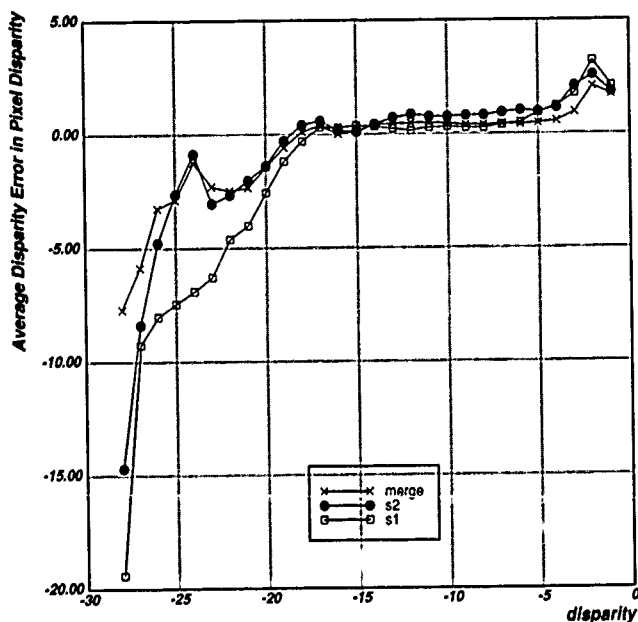


Figure 4-5: Average Error in Pixel Disparity at Each Disparity Level in Denver ALV

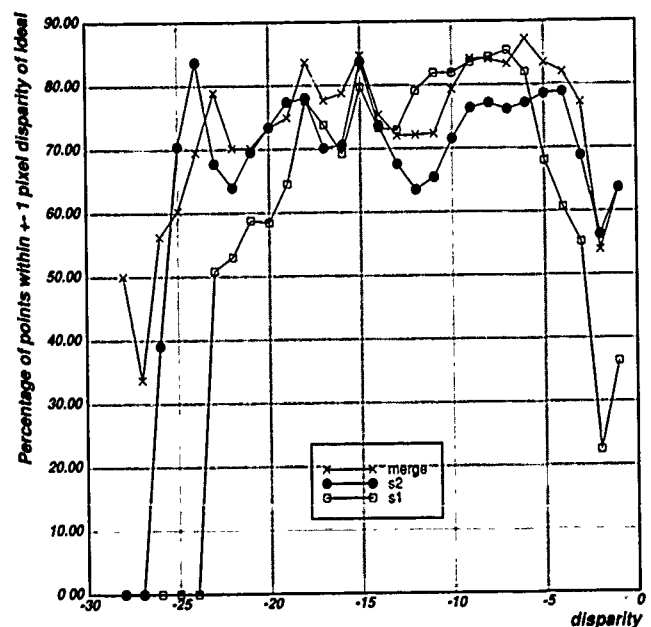


Figure 4-6: Percentage of Points within ± 1 Pixel of Ideal Disparity in Denver ALV

result, although S2 appears to perform better, especially on buildings with greater disparity. For most buildings the error is bounded between \pm two pixels. The result of merging generally appears to improve the average error. As we have assigned building ID's sorted by disparity we can observe a trend towards increased error as we move along the x-axis.

We can also represent results using the disparity jump instead of the building ID to index the results. These graphics represent the integration of the average disparity error over all buildings with the same disparity jump. Figure 4-12 and 4-13 show the effect of disparity jump on the disparity estimate and allow us to determine whether the actual height of a building

over its neighborhood (disparity jump) affects the disparity estimate produced by stereo matching. It appears that S1 is comparable with S2 for smaller buildings. This is because low buildings can satisfy the continuity constraint of the area-based method. S2 performs better on scenes with buildings having significant height because low buildings can be easily masked by random mismatches in the feature-based analysis. The merge of S1 and S2 produces results that combine the best properties of both methods.

Figures 4-14, 4-15 and 4-16 provide similar statistics for the suburban house scene, DC37405. As in DC38008 the average error for each building appears to be bounded by \pm two

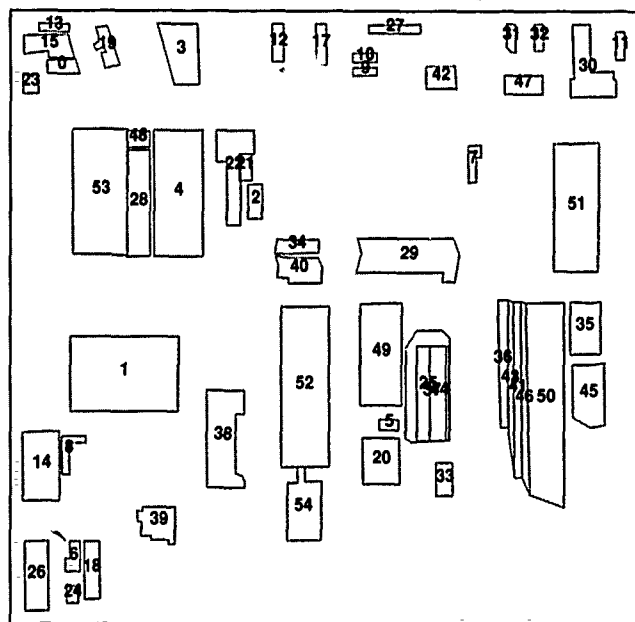


Figure 4-7: Building Index for DC38008

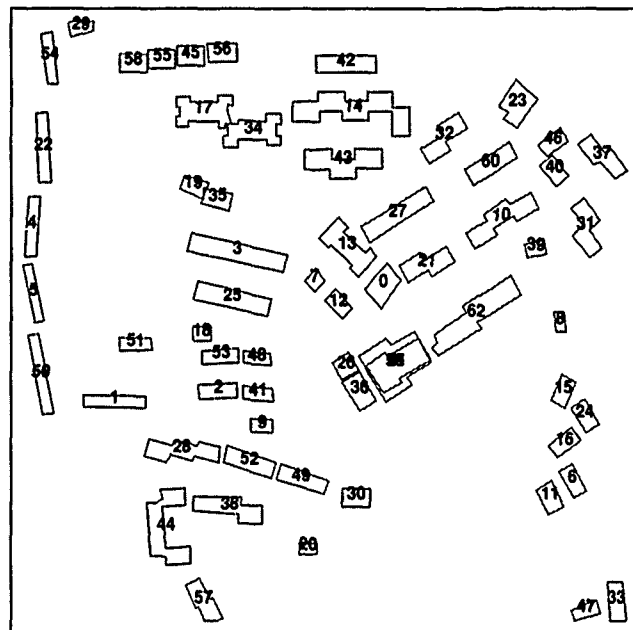


Figure 4-8: Building Index for DC37405

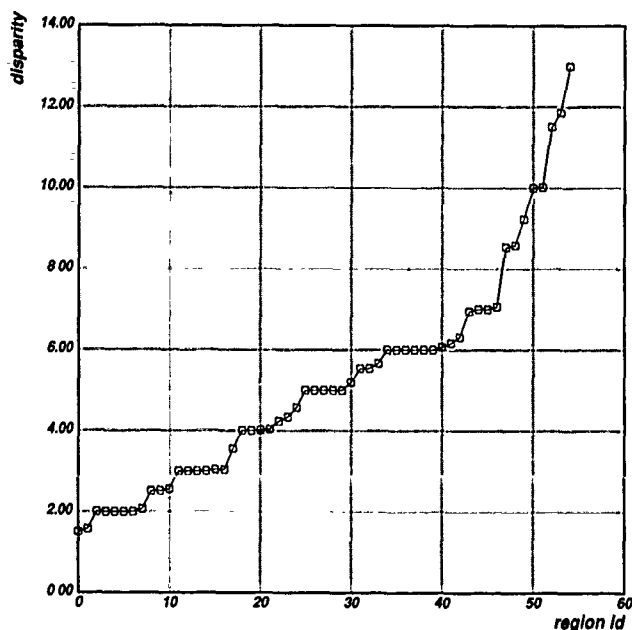


Figure 4-9: Building Heights for Figure 4-7

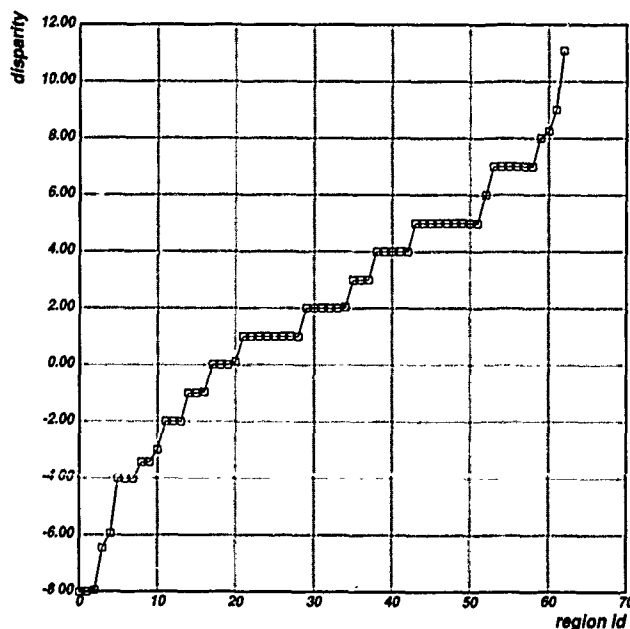


Figure 4-10: Building Heights for Figure 4-8

pixels, S2 appears to have slightly better performance than S1, and the result of the merger almost always improves the average error. Whereas S2 always appears to perform much better than S1 with respect to the percentage points (within ± 1 pixel of the correct disparity in DC38008), (Figure 4-13) this is not the case for DC37405 as shown in Figure 4-16.

These statistics allow us to pinpoint problems at a much finer grain of detail than can be accomplished with global analysis. Thus we can identify specific buildings in the scene and try to understand, at the algorithmic level, whether there are specific

situations where matching could be improved. Once identified, these improvements should have an overall positive effect on the rest of the scene. The result, of course, can be subjected to the same rigorous performance analysis. Once we commit to working on complex scenes, as opposed to synthetic controlled images, the visual inspection of disparity results to discover small variations in performance becomes very unsatisfactory, except possibly at the earliest stages of experimentation. Such manual inspection greatly limits our ability to detect subtle conceptual bugs or recognize possibilities for algorithmic improvement. In some cases we can perform systematic

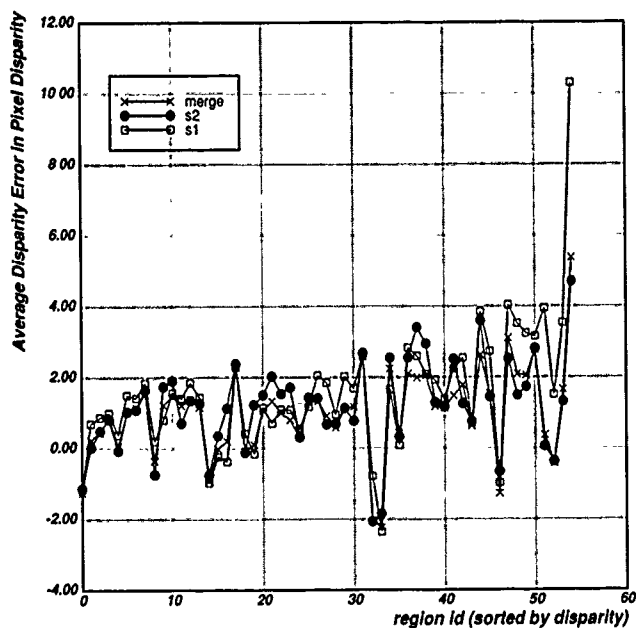


Figure 4-11: Average Error for Each Building in DC38008

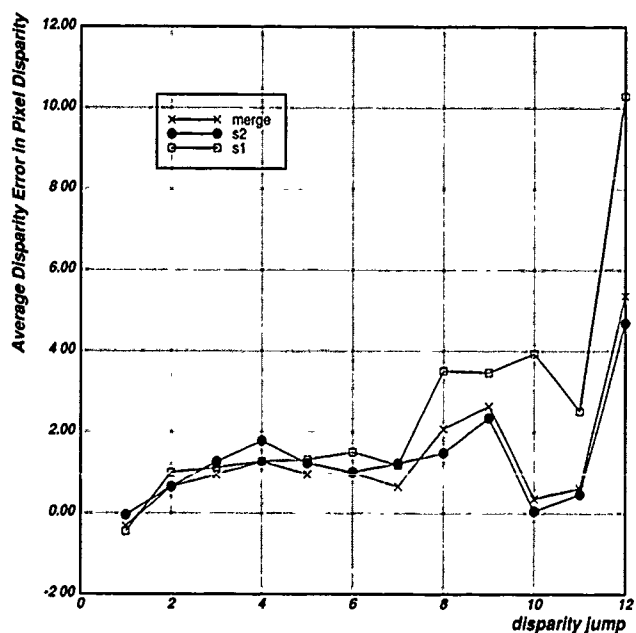


Figure 4-12: Average Error for Each Disparity Jump in DC38008

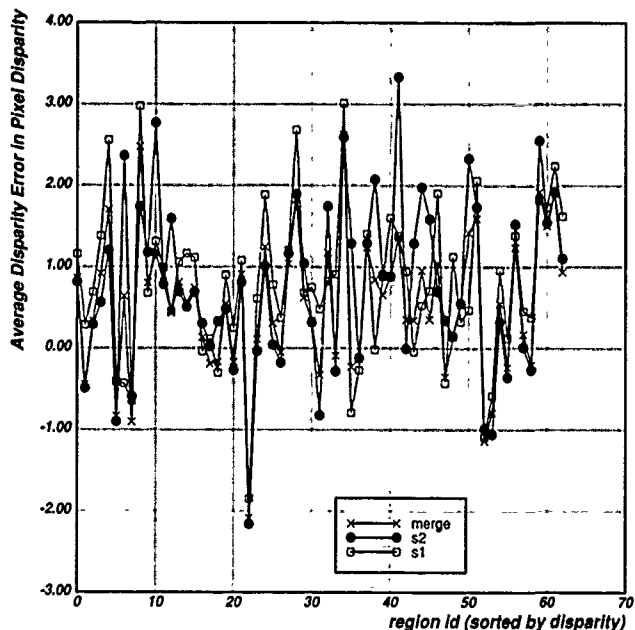


Figure 4-14: Average Error for Each Building in DC37405

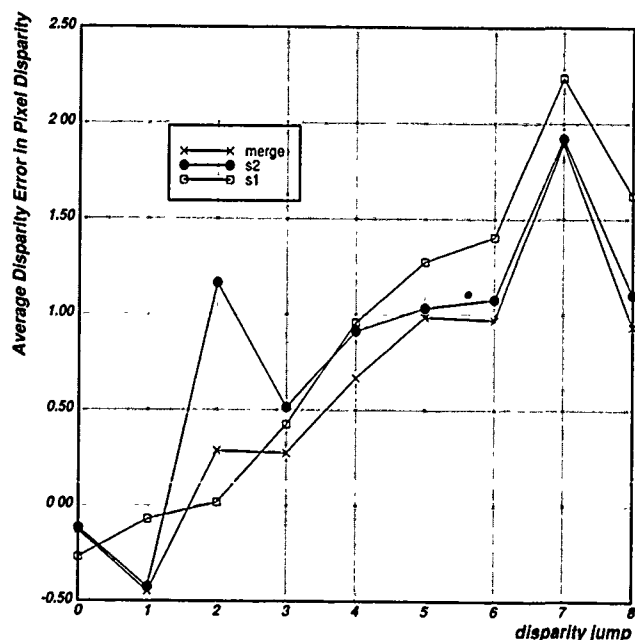


Figure 4-15: Average Error for Each Disparity Jump in DC37405

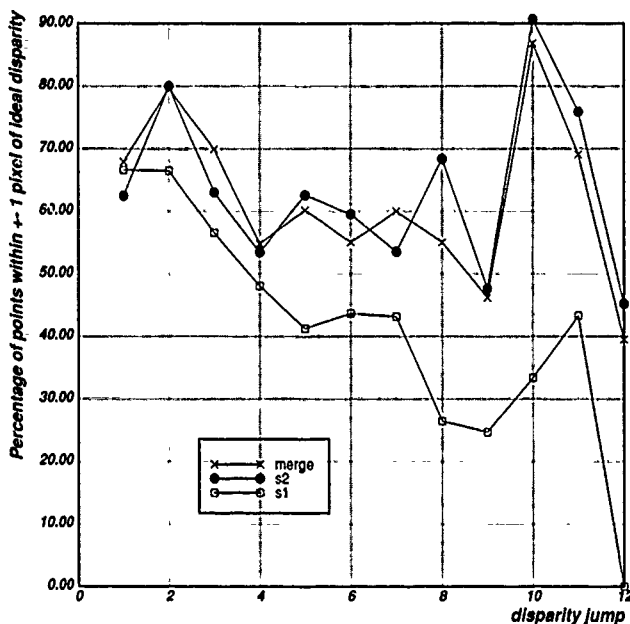


Figure 4-13: Percentage of Good Points for Each Disparity Jump in DC38008

analysis across multiple scenes. For example, in applying statistics that take into account the disparity jump for individual buildings, we can aggregate performance information for all buildings across all scenes to achieve a larger statistical sample.

4.2. Quality of Delineation Estimate

In the previous section we described techniques to measure the accuracy with which we can recover the height of buildings in the scene. For cartographic applications it is equally important that we generate an accurate delineation of the buildings with respect to their surroundings. In this section we discuss another metric which is the quality of the stereo delineation of each building in the scene. We compute *edge location* which measures the distance of the estimated disparity jump from that in the ground-truth disparity. We also measure *edge sharpness* which corresponds to the shape of the disparity jump in the estimated disparity map. Ideally, we would expect the stereo matcher to generate a step disparity jump at the point where the actual disparity jump occurs in the reference disparity map. As before, we assume that the ground-truth disparity map accurately captures the location and the height of the building edges. In order to allow for measurement error, we tolerate some uncertainty in both the location of the edge (\pm one pixel) and the height estimate on both sides of the edge (edge sharpness). The uncertainty in edge sharpness is somewhat difficult to quantify since it depends on both the height estimate on each side of the building roof edge and on the height estimate of the neighboring ground. These estimates may be biased, since in some cases we are interpolating the ground elevation from a sparse network of points. We can alleviate this error by making sure that we select representative ground points as close to the buildings as possible.

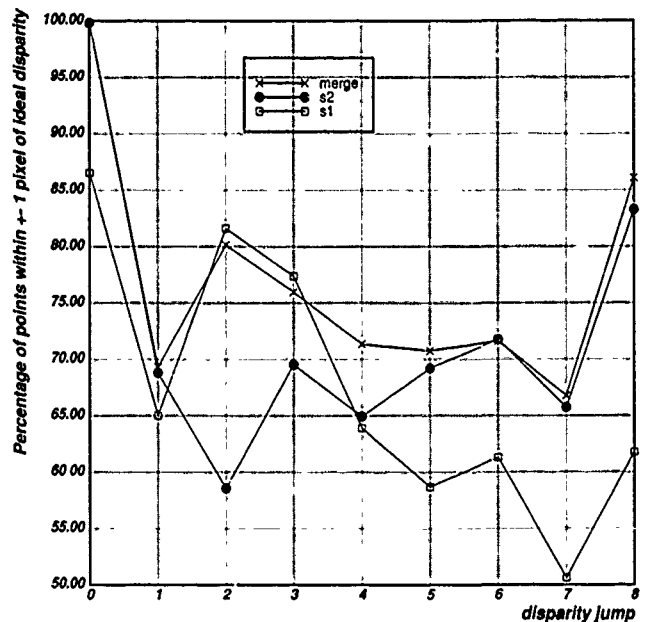


Figure 4-16: Percentage of Good Points for Each Disparity Jump in DC37405

Figure 4-17 shows how we compute the edge location and sharpness for each building in the scene. The two waveforms represent the gradient of the reference disparity map and the disparity result being evaluated. The peaks in the reference disparity map gradient represent the true edge of the building in the scene. The evaluation process finds the best matching peaks in the S1, S2, or merged disparity map gradient within a neighborhood of the reference edge. The distance P corresponds to the position error of the edge in the result disparity map. The ratio H_d/H_r corresponds to the sharpness evaluation of the edge. A ratio of one is perfect. The value H_d and H_r correspond to the amplitude of the gradient related to the reference zero gradient.

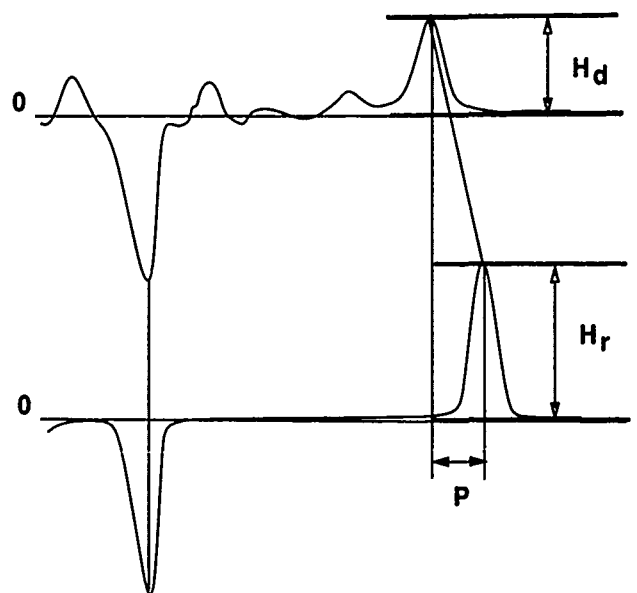


Figure 4-17: Gradient Matching for Edge Evaluation

Both the position error and the edge sharpness metric require that an edge point in the reference disparity map be matched with an edge point produced by the stereo matcher under evaluation. In many cases no such match is possible; that is, there is no suitable match for the reference disparity edge. In the following examples between 35% (DC37405) and 50% (DC38008) of the reference points are not matched, hence the *matchable* edges represent between 50-65% of the reference points in the scene. Figures 4-18 and 4-20 represent the average position error for the *matchable* edges across all buildings in DC38008 and DC37405, respectively.

Figures 4-19 and 4-21 shows the percentage of edges produced by the stereo matchers that are within ± 1 pixel of a reference disparity map edge. These graphs are the subset of points lying in the band ± 1 position error from Figures 4-18 and 4-20 respectively, plotted with respect to all edges in the reference disparity map. In both cases the position error metric shows that the ability to accurately delineate the disparity depth jump appears to be much weaker than visual examination of the disparity maps might indicate.

For the evaluation of disparity sharpness we calculate the average edge ratio and the sharpness of edge points whose edge

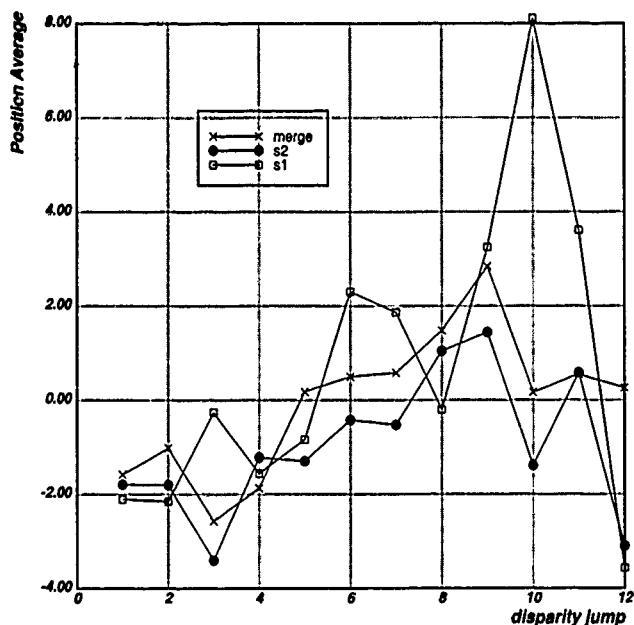


Figure 4-18: Edge Position Error for DC38008

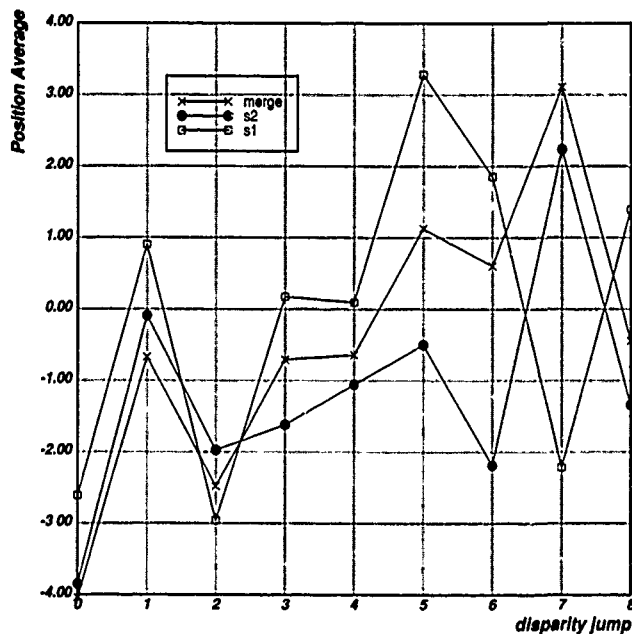


Figure 4-20: Edge Position Error for DC37405

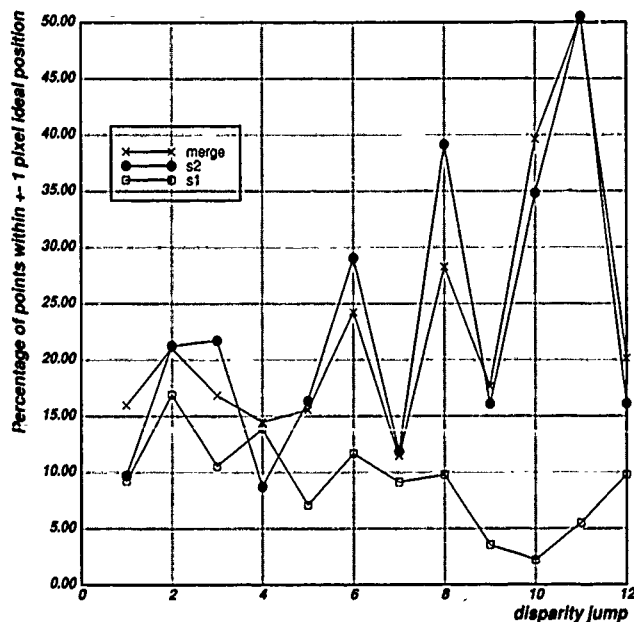


Figure 4-19: Percent Good Edgels for DC38008

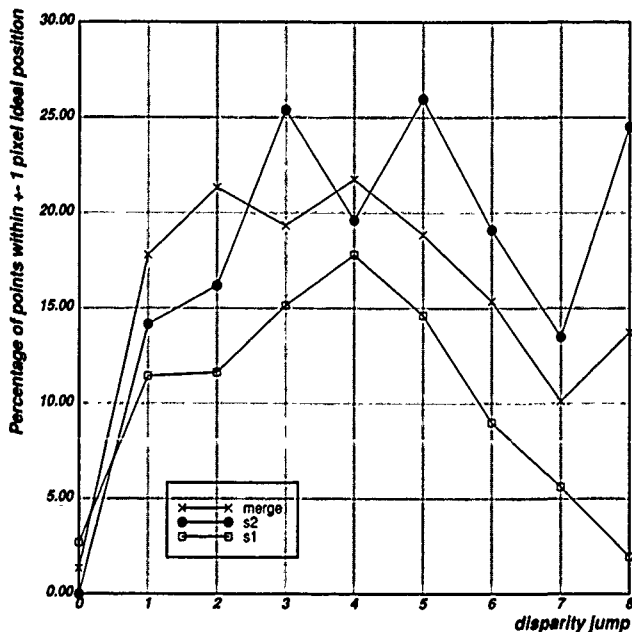


Figure 4-21: Percent Good Edgels for DC37405

position is within \pm one pixel of the reference edge. Figure 4-22 represents the average edge sharpness ratio for all matchable edges across all buildings in DC38008. A ratio of one indicates a perfect step edge. Figure 4-23 shows the sharpness of edge points that are within \pm one pixel of the reference position for all buildings in DC38008. Figures 4-24 and 4-25 show the same results applied to the buildings in DC37405.

We can make several observations based upon this performance data. First, it is clear from this analysis that S1 does not perform as well as S2 in terms of disparity delineation. Its ability to estimate the sharpness of the disparity jump (edge

ratio) is likewise poorer than that of S2. However, there are some comparative advantages. S1 gives comparable results in the case of buildings with low disparity. On the DC37405 scene the S1 and S2 results are similar because the buildings in this scene do not have large disparity jumps.

It is interesting to note that errors in delineation, position, and sharpness increase as the height of the buildings increase. This is an artifact of occlusion, where higher buildings will occlude a larger area, making it more difficult to detect the exact position of the disparity jump. Edge errors seem to be

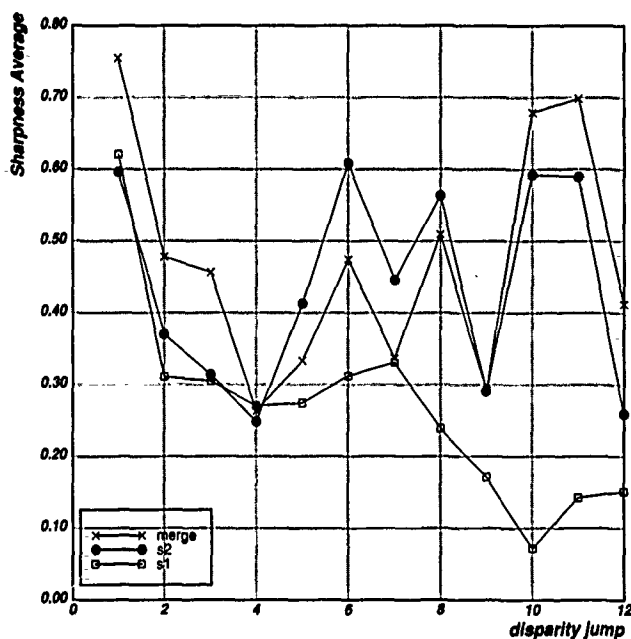


Figure 4-22: Sharpness for DC38008

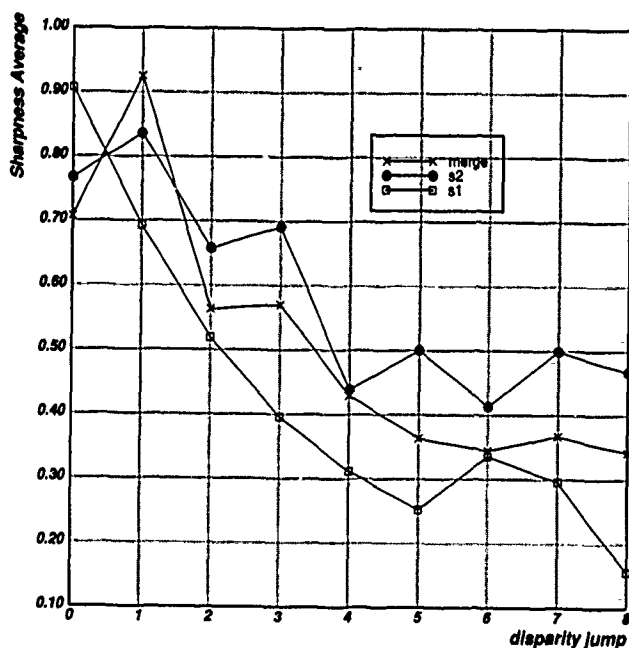


Figure 4-24: Sharpness for DC37405

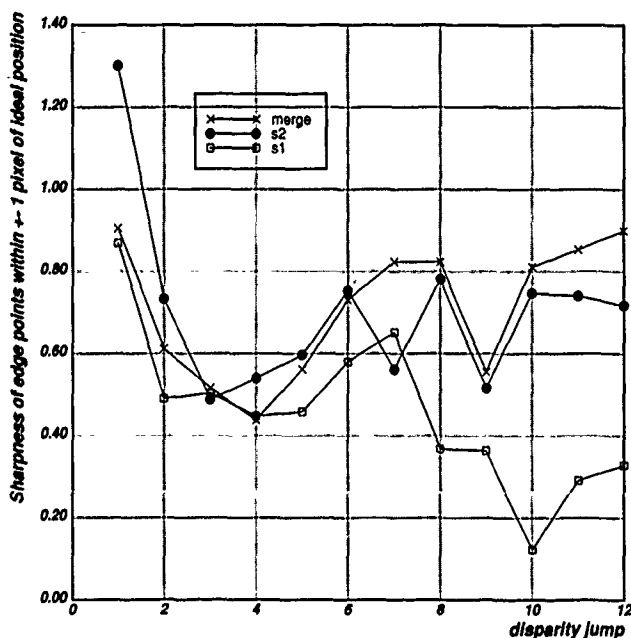


Figure 4-23: Sharpness of Good Edgels for DC38008

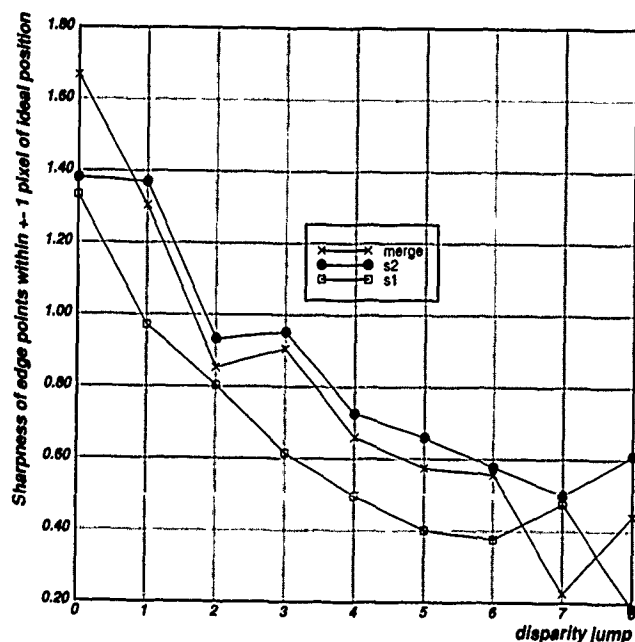


Figure 4-25: Sharpness of Good Edgels for DC37405

comparable for both S1 and S2 for buildings with low disparity. As expected, S1 does not delineate tall buildings well and the merged result combining S1 and S2 sometimes produces a result that is an improvement over each individual method but, more often, simply decreases the maximal error.

4.3. Limitations of Performance Evaluation

The common theme in this section on performance evaluation is to describe a variety of quantitative measures that allow us to objectively judge how well a particular set of registration/matching techniques perform with respect to a manually compiled three-dimensional ground-truth model, and by comparison, how well they perform with respect to one another. The reference disparity map is generated using monocular and stereoscopic visualization and is a representation of the scene within a certain accuracy. In most cases the ground-truth segmentation can be constructed with enough care to provide for accurate detection of gross errors, and as a common basis for general comparison between matching methods. However, the actual accuracy of the reference disparity map has to be considered if we attempt to use it for the analysis of scene micro-structure, such as roofs with shallow pitch that are modeled as flat surfaces, small super structures such as building air conditioner units, stair well towers, and other small roof structures. These superstructures can add an error bias into the overall statistics. This bias is likely to be small; consider the fraction of error introduced in the case of a nine story building where we have not correctly modeled an air conditioner unit that rises another story over 15% of the total roof surface.

Nevertheless, we are sampling only a small subset of the actual three-dimensional points in the scene. If we count all of the building edge pixels and terrain web points manually selected for scenes such as DC38008 and DC37405, less than 3% of the scene points are used to produce the dense reference disparity map. These points are represented in a triangulated irregular network (TIN) for the terrain upon which is superimposed the building roof structures. We linearly interpolate the network in order to calculate the dense disparity map. Interestingly, S2 gives us matches for approximately 12% of the scene points which is typical for feature-based matching algorithms. As such, our performance analysis is subject to possible errors in the evaluation of S2 matching algorithm introduced due to interpolation from the sparse disparity map.

Given the lack of performance evaluation techniques in computer vision for three-dimensional scene modeling we are probably content simply to know the height of the buildings and the general shape of the underlying terrain. But we should understand that if we attempt to push performance analysis to detail the small effects of subtle algorithmic changes we may run up against fundamental limits in our ability to recover these micro-structures. Thus, in our calculations, we have added an uncertainty of \pm one pixel of disparity to the ideal ground-truth value and feel that this covers a large fraction of the

inherent inaccuracies. In summary, our disparity performance evaluation has to be considered as a method to easily detect large mismatches by the stereo analysis system; it may have some limitations in the fine evaluation of disparity values. Nevertheless, we see such techniques as the only method for effective comparison of disparity results.

5. Conclusions

Fully automated stereo analysis in complex urban scenes is a difficult research problem. In this paper we have discussed three major areas in the development of competent three-dimensional scene interpretation systems. First, we discussed the importance of accurate automatic scene registration and the difficulty in automated extraction and matching of scene reference points. We showed several results in fully automated scene registration including the estimation of the scene disparity range as a necessary parameter for stereo matching algorithms.

Secondly, we described two stereo matching algorithms: S1, an area-based matcher previously used in the SPAM system¹⁴; and S2, a new feature-based matching algorithm based upon hierarchical waveform matching. We also introduced a technique to merge the results of the two matching algorithms which appears to give an improved disparity map and also indicates areas where occlusion and gross mismatches may have occurred.

Finally, we introduced several performance evaluation metrics that allowed us to measure the quality of the overall scene recovery, the building disparity estimate, and the quality and sharpness of the building delineations. We argue that such manually generated scene reference models are critical for understanding strengths and weaknesses of various matching algorithms, and in the incremental development of improvements to existing algorithms.

We performed these experiments on difficult examples of aerial imagery containing complex urban scenes with variations in terrain, building shape, size, and height, as well as in an example of open terrain.

Our future work is directed toward improvements in the basic S1 and S2 matching algorithms, the refinement of our ground-truth disparity maps to allow for a finer detail of analysis, and in techniques that will allow us to merge and refine our three-dimensional scene interpretations using information available from monocular analysis of the scene.

6. Acknowledgments

We thank our colleagues in the Digital Mapping Laboratory for an interesting and congenial working environment. We particularly acknowledge Wilson Harvey and Jeff Shufelt who helped in the preparation of this paper, and Emily Burke who organized our racquetball schedules.

7. Bibliography

1. Barnard, S. T. and Fischler, M. A., "Computational stereo," *Computing Surveys*, Vol. 14, No. 4, December 1982, pp. 553-572.
2. R.D. Arnold, "Local Context in Matching Edges for Stereo vision," *Proceedings: DARPA Image Understanding Workshop*, May 1978, pp. 777-791.
3. S. T. Barnard, "Stochastic stereo matching over scale," *Proceedings: DARPA Image Understanding Workshop*, April 1988, pp. 769-778.
4. N. M. Nasrabadi, Y. Liu, and J-L. Chiang, "Stereo vision correspondence using a multi-channel graph matching technique," *IEEE International Conference on Robotics and Automation*, April 1988.
5. Y. Ohta and T. Kanade, "Stereo by Intra- and Inter-scanline Search using Dynamic Programming," *IEEE Transactions*, Vol. PAMI-7, No. 2, March 1985, pp. 139-154.
6. McKeown, D.M., "Digital Cartography and Photo Interpretation from a Database Viewpoint," in *New Applications of Databases*, Gargarin, G. and Golembe, E., ed., Academic Press, New York, N. Y., 1984, pp. 19-42.
7. McKeown, D.M., "The Role of Artificial Intelligence in the Integration of Remotely Sensed Data with Geographic Information Systems," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. GE-25, No. 3, May 1987, pp. 330-348.
8. H. P. Moravec, *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*, PhD dissertation, Stanford University, September 1980.
9. Perlant, F. P., McKeown, D.M., "Scene Registration in Aerial Image Analysis," *Photogrammetric Engineering and Remote Sensing*, Vol. Volume 56, No. 4, April 1990, pp. 481-493.
10. Aviad, Z., McKeown, D. M., Hsieh, Y., "The Generation of Building Hypotheses From Monocular Views," Tech. report, Carnegie-Mellon University, 1990, to appear
11. R. B. Irvin and D. M. McKeown, "Methods for exploiting the relationship between buildings and their shadows in aerial imagery," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 19, No. 6, November 1989, pp. 1564-1575.
12. McKeown, D.M. and Denlinger, J. L., "Cooperative Methods for Road Tracking in Aerial Imagery," *Proceedings IEEE Computer Vision and Pattern Recognition Conference*, June 1988, pp. 662-672.
13. B. D. Lucas, *Generalized Image Matching By The Method of Differences*, PhD dissertation, Carnegie Mellon University, July 1984.
14. McKeown, D.M., McVay, C.A., and Lucas, B. D., "Stereo Verification In Aerial Image Analysis," *Optical Engineering*, Vol. 25, No. 3, March 1986, pp. 333-346.
15. Cochran, S.D. and Medioni, G., "Accurate surface description from binocular stereo," *Image Understanding Workshop*, DARPA, Palo Alto, CA., May 1989, pp. 857-869.
16. Witkin, A., Terzopoulos, D. and Kass, M., "Signal Matching Through Scale Space," *International Journal of Computer Vision*, Vol. , No. , 1987, pp. 133-144.
17. Baker, H.H. and Binford, T.O., "Depth from edge and intensity based stereo," *Proc. 7th International Conference on Artificial Intelligence*, August 1981, pp. 631-636.
18. Cheng, Y.-C. and LU, S.-Y., "Waveform correlation by tree matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 7, No. 3, May 1985, pp. 299-305.
19. Ballard, D. H. and Brown, C. M., *Computer Vision*, Prentice-Hall, Englewood Cliffs, NJ, 1982.
20. Hoff, W. and Ahuja, N., "Surface from stereo: integrating feature matching, disparity estimation and contour detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 2, February 1989, pp. 121-136.
21. Boulton, T.E. and Chen, L.-H., "Synergistic smooth surface stereo," *IEEE International Conference on Computer Vision*, IEEE, Tampa, Florida, December 1988, pp. 118-122.
22. Grimson, W., *From Images to Surfaces: A Computational Study of the Human Early Visual System*, The MIT Press, Cambridge, Massachusetts, 1981.

Fusion of Monocular Cues to Detect Man-Made Structures in Aerial Imagery

Jefferey Shufelt and David M. McKeown

Digital Mapping Laboratory
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213¹

Abstract

The extraction of buildings from aerial imagery is a complex problem for automated computer vision. It requires locating regions in a scene that possess properties distinguishing them as man-made objects as opposed to naturally occurring terrain features. The building extraction process requires techniques that exploit knowledge about the structure of man-made objects. Techniques do exist that take advantage of this knowledge; various methods use edge-line analysis, shadow analysis, and stereo imagery analysis to produce building hypotheses. It is reasonable, however, to assume that no single detection method will correctly delineate or verify buildings in every scene. As an example, a feature extraction system that relies on the analysis of cast shadows to predict building locations is likely to fail in cases where the sun is directly above the scene.

It seems clear that a cooperative-methods paradigm is useful in approaching the building extraction problem. Using this paradigm, each extraction technique provides information which can then be added or assimilated into an overall interpretation of the scene. Thus, our research focus is to explore the development of a computer vision system that integrates the results of various scene analysis techniques into an accurate and robust interpretation of the underlying three-dimensional scene.

This paper describes preliminary research on the problem of building hypothesis fusion in aerial imagery. Building extraction techniques are briefly surveyed, including four building extraction, verification, and clustering systems that form the basis for the work described here. A method for

fusing the symbolic data generated by these systems is described, and applied to monocular image and stereo image data sets. Evaluation methods for the fusion results are described, and the fusion results are analyzed using these methods.

1. Introduction

In the cooperative-methods paradigm it is assumed that no single method can provide a complete set of building hypotheses for a scene. However, each method may provide a subset of the information necessary to produce a more meaningful interpretation of the scene. For instance, a shadow-based method might provide unique information in situations where ground and roof intensity are similar. An intensity-based method can provide boundary information in instances where shadows were weak or nonexistent, or in situations where structure height was sufficiently low that stereo disparity analysis would not provide reliable information. The implicit assumption behind this paradigm is that the symbolic interpretations produced by each of these techniques can be integrated into a more meaningful collection of building hypotheses.

It is reasonable to expect that there will be complications in fusing real monocular data. In the best case, the building hypotheses will not only be accurate, but complementary. It is just as likely, however, that some building hypotheses may be unique. Further, it is rare that building hypotheses are always accurate, or even mutually supportive of one another. For a cooperative-methods data fusion system to be successful, it must address the problems of redundant and conflicting data.

2. Building extraction techniques

At the Digital Mapping Laboratory, we have developed several techniques for the extraction of man-made objects from aerial imagery. The goal of many of these techniques is to organize the image into manageable parts for further processing, by using external knowledge to organize these parts into regions.

¹This research was primarily sponsored by the U.S. Army Engineer Topographic Laboratories under Contract DACA72-87-C-0001 and partially supported by the Defense Advanced Research Projects Agency, DoD, through DARPA order 4976, and monitored by the Air Force Avionics Laboratory Under Contract F33615-87-C-1499. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Engineering Topographic Laboratories, or the Defense Advanced Research Projects Agency, or of the United States Government.

For the experiments described in this paper, a set of four monocular building detection and evaluation systems were used. Three of these were shadow-based systems; the fourth was line-corner based. The shadow based systems are described more fully by Irvin and McKeown [7], and the line-corner system is described by Aviad, McKeown, and Hsieh [2]. A brief description of each of the four detection and evaluation systems follows.

BABE (Builtup Area Building Extraction) is a building detection system based on a line-corner analysis method. BABE starts with intensity edges for an image, and examines the proximity and angles between edges to produce corners. To recover the structures represented by the corners, BABE constructs chains of corners such that the direction of rotation along a chain is either clockwise or counterclockwise, but not both. Since these chains may not necessarily form closed segmentations, BABE generates building hypotheses by forming boxes out of the individual lines that comprise a chain. These boxes are then evaluated in terms of size and line intensity constraints, and the best boxes for each chain are kept, subject to shadow intensity constraints [6], [10].

SHADE (SHadow DETection) is a building detection system based on a shadow analysis method. SHADE uses the shadow intensity computed by BABE as a threshold for an image. Connected region extraction techniques are applied to produce segmentations of those regions with intensities below the threshold, i.e., the shadow regions. SHADE then examines the edges comprising shadow regions, and keeps those edges that are adjacent to the buildings casting the shadows. These edges are then broken into nearly straight line segments by the use of an imperfect sequence finder [1]. Those line segments that form nearly right-angled corners are joined, and the corners that are concave with respect to the sun are extended into parallelograms, SHADE's final building hypotheses.

SHAVE (SHadow VERification) is a system for verification of building hypotheses by shadow analysis. SHAVE takes as input a set of building hypotheses, an associated image, and a shadow threshold produced by BABE. SHAVE begins by determining which sides of the hypothesized building boxes could possibly cast shadows, given the sun illumination angle, and then performs a walk away from the sun illumination angle for every pixel along a building/shadow edge to delineate the shadow. The edge is then scored based on a measure of the variance of the length of the shadow walks for that edge. These scores can then be examined to estimate the likelihood that a building hypothesis corresponds to a building, based on the extent to which it casts shadows.

GROUPEr is a system designed to cluster, or group, fragmented building hypotheses, by examining their relationships to possible building/shadow edges. GROUPEr starts with a set of hypotheses and the building/shadow edges produced by BABE. GROUPEr back-projects the endpoints of a building/shadow edge towards the sun along the sun illumination angle, and then connects these projected endpoints to form a region of interest in which buildings might occur. GROUPEr intersects each building hypothesis with these regions of interest. If the degree of overlap is sufficiently high (the criteria is currently 75% overlap), then the hypothesis is assumed to be a part of the structure which is casting the building/shadow edge. All hypotheses that intersect a single

region of interest are grouped together to form a single building cluster.

There are many other interesting building detection and extraction techniques in the contemporary literature. We briefly mention some recently developed methods, to illustrate the variety of techniques that produce building hypothesis information. Mohan and Nevatia [9] described a method by which simple image tokens such as lines or edges could be clustered into more complex geometric features consisting of parallelepipeds. Huertas and Nevatia [6] described a method for detecting buildings in aerial images. Their method detected lines and corners in an image and constructed chains of these to form building hypotheses which were then subject to shadow verification.

Fua and Hanson [3] described a system that used generic geometric models and noise-tolerant geometry parsing rules to allow semantic information to interact with low-level geometric information, producing segmentations of objects in the aerial image. Nicolin and Gabler [10] described a system for analysis of aerial images. The system had four components: a method-base of domain-independent processing techniques, a long-term memory containing *a priori* knowledge about the problem domain, a short-term memory containing intermediate results from the image analysis process, and a control module responsible for invocation of the various processing techniques. Gray-level analysis was applied to a resolution pyramid of imagery to suggest segmentation techniques, and structural analysis was performed after segmentation to provide geometric interpretations of the image.

Thus, there is a fairly rich set of building extraction systems, each with its own particular strengths and weaknesses. Although this by no means constitutes a comprehensive survey of building detector techniques, it provides some examples of the methods used to generate hypotheses for a scene, as well as examples of the types of data that may eventually be integrated into a cooperative-methods building analysis scheme.

3. A simple hypothesis merging technique

Building hypotheses typically take the form of geometric descriptions of objects in the context of an image. One can imagine "stacking" sets of these geometric descriptions on the image: in the process, those regions of the image that represent man-made structure in the scene should accumulate more building hypotheses than those regions of the image that represent natural features in the scene. The merging technique developed here exploits this idea.

The method takes as input an arbitrary collection of polygons. An image is created that is sufficiently large to contain all of the polygons, and each pixel in this image is initialized to zero. Each polygon is scan-converted into the image, and each pixel touched during the scan is incremented. The resulting image then has the property that the value of each pixel in the image is the number of input polygons that cover it.

Segmentations can then be generated from this "accumulator" image by applying connected region extraction techniques. If the image is thresholded at a value of 1 (i.e., all non-zero pixels are kept), the regions produced by a connected region extraction algorithm will simply be the geometric unions of the input polygons. It is the case, however, that the

image could be thresholded at higher values. We motivate thresholding experiments in Section 4.4.

4. Merging multiple hypothesis sets

This section outlines the experiments performed with the scan-conversion hypothesis fusion technique. The procedure used to apply this technique to the results of four building detection and evaluation systems (BABE, SHADE, SHAVE, and GROUPER) is described. A technique for quantitative evaluation of building hypotheses is described, and applied to the hypothesis fusion results. These results are analyzed to suggest improvements to the fusion technique.

4.1. The merging technique applied to four extraction systems

There were two merging problems under consideration. The first of these was the creation of a single hypothesis out of a collection of fragmented hypotheses believed to correspond to a single man-made structure. This problem was addressed by

applying the scan-conversion technique to the fragmented clusters produced by GROUPER. The technique was applied to each cluster individually, and the resulting accumulator image was thresholded at 1, and connected region extraction techniques were applied to provide the geometric union of each cluster. These clusters were then used as the building hypotheses produced by GROUPER.

The second problem was the fusion of each of these monocular hypothesis sets into a single set of hypotheses for the scene. Again, the scan-conversion technique was applied. The four hypothesis sets were scan-converted, and the resulting accumulator image was thresholded at 1. Connected region extraction techniques were applied to produce the final segmentation for the image.

Figure 4-1 shows a section of a suburban area in Washington, D.C. Figure 4-2 shows the SHADE results for this scene. Figure 4-3 shows the SHAVE results. Figure 4-4 shows the GROUPER results, and Figure 4-5 shows the BABE results. Figure 4-6 shows the fusion of these four monocular hypothesis sets.

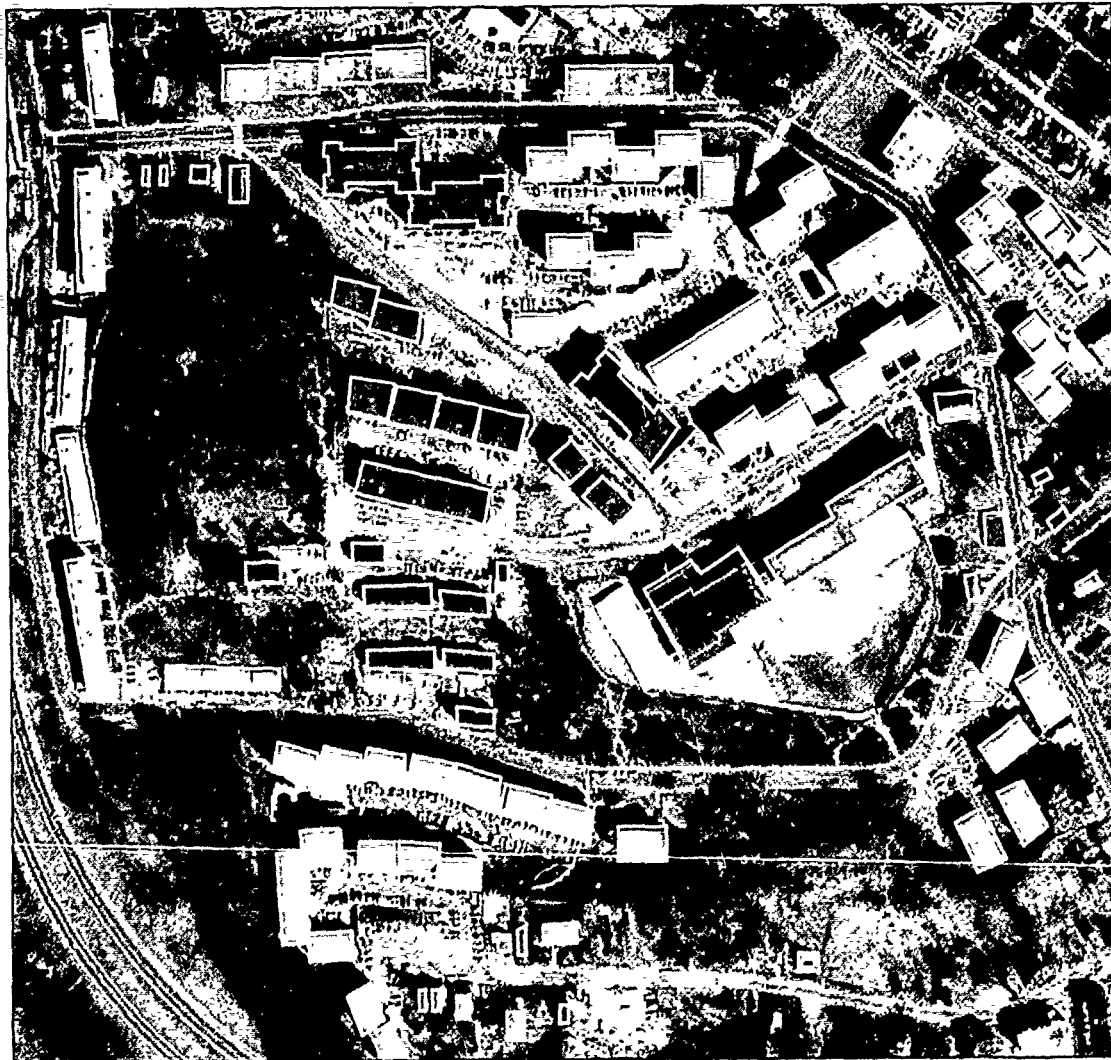


Figure 4-1: DC37 image with ground-truth segmentation

4.2. Evaluation of the technique

To judge the correctness of an interpretation of a scene, it is desirable to have some mechanism for quantitatively evaluating that interpretation. One approach is to compare a given set of hypotheses against a set that is known to be correct, and analyze the differences between the given set of hypotheses and the correct ones. In performing evaluations of the fusion results, we use *ground-truth segmentations* as the correct detection results for a scene. Ground-truth segmentations are manually produced segmentations of the buildings in an image.

There exist two simple criteria for measuring the degree of similarity between a building hypothesis and a ground-truth building segmentation: the mutual area of overlap and the difference in orientation. A correct building hypothesis and the

corresponding ground-truth segmentation region should cover roughly the same area, and should have roughly the same alignment with respect to the image. A scoring function can be developed that incorporates these criteria. A region matching scheme such as this, however, suffers from the fact that multiple buildings in the scene are segmented by a single region in the hypothesis set. In these cases, the building hypothesis will have low matching scores with each of the buildings it contains, due to the differences in overlap area.

A simpler coverage-based global evaluation method was developed. This evaluation method works in the following manner. H , a set of building hypotheses for an image, and G , a ground-truth segmentation of that image, are given. The image is then scanned, pixel by pixel. For any pixel P in the image, there are four possibilities:



Figure 4-2: DC37 SHADE results



Figure 4-4: DC37 GROUPEUR results



Figure 4-3: DC37 SHAVE results



Figure 4-5: DC37 BABE results

1. Neither a region in H nor a region in G covers P . This is interpreted to mean that the system producing H correctly denoted P as being part of the background, or natural structure, of the scene.
2. No region in H covers P , but a region in G covers P . This is interpreted to mean that the system producing H did not recognize P as being part of a man-made structure in the scene. In this case, the pixel is referred to as a "false negative".
3. A region (or regions) in H cover P , but no region in G covers P . This is interpreted to mean that the system producing H incorrectly denoted P as belonging to some man-made structure, when it is in fact part of the scene's background. In this case, the pixel is referred to as a "false positive".
4. A region (or regions) in H and a region in G both cover P . This is interpreted to mean that the system producing H correctly denoted P as belonging to a man-made structure in the scene.

By counting the number of pixels that fall into each of these four categories, we may obtain measurements of the percentage of building hypotheses that were successful (and unsuccessful) in denoting pixels as belonging to man-made structure, and the percentage of the background of the scene that was correctly (and incorrectly) labeled as such. Further, we may use these measurements to define a *building pixel branching factor*, which will represent the degree to which a building detection system overclassifies background pixels as building pixels in the process of generating building hypotheses. The building pixel branching factor is defined as the number of false positive pixels divided by the number of correctly detected building pixels.

4.3. Results and analysis

The fusion process was run on other scenes in addition to the DC37 scene: DC36A, DC36B, and DC38, three more scenes from the Washington, D.C. area; and LAX, a scene from the Los Angeles International Airport [6]. The final fusion results for each of these scenes are shown in Figures 4-7 through 4-10. The coverage-based evaluation program was then applied to each of these results to generate Tables 4-1 through 4-5. Each

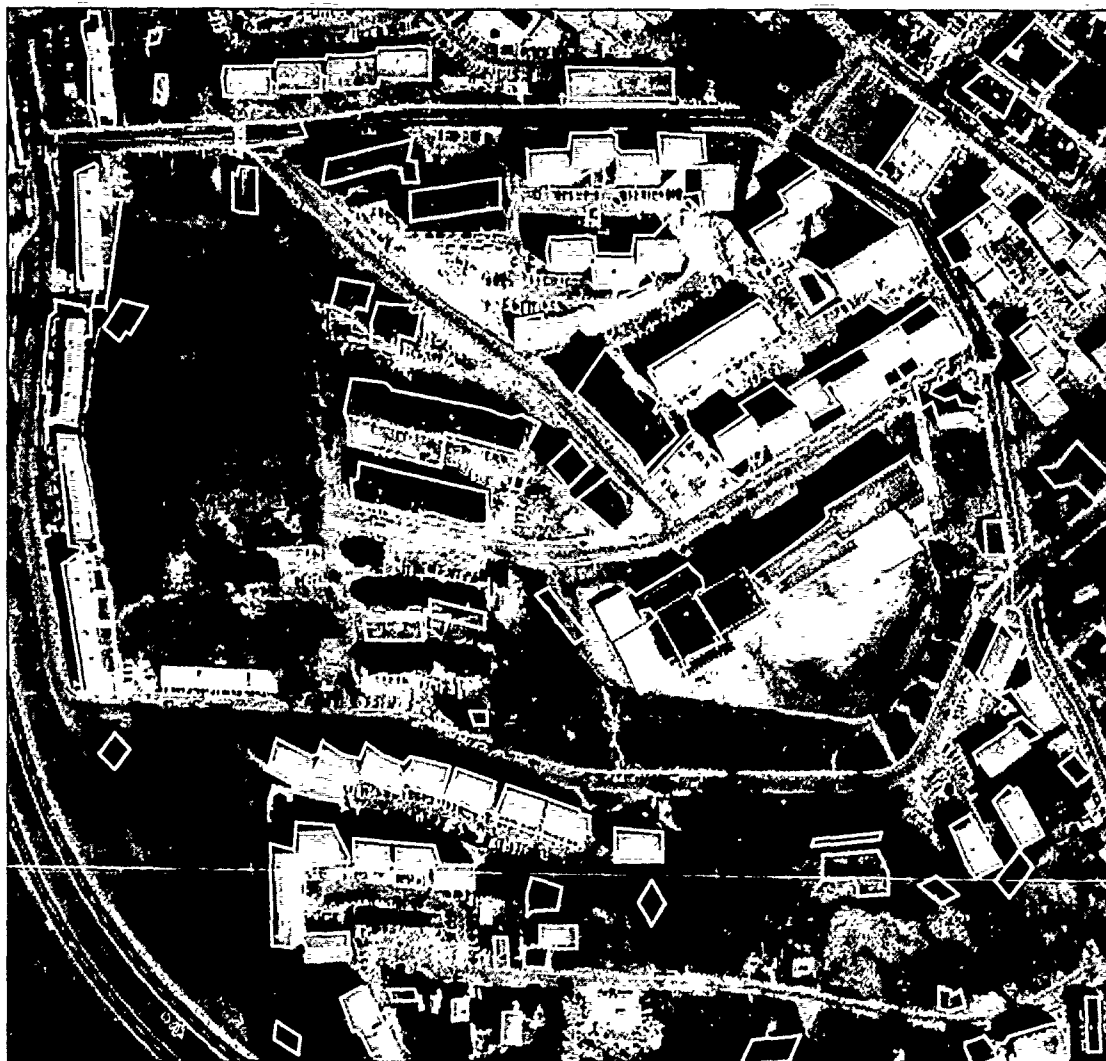


Figure 4-6: Monocular hypothesis fusion for DC37

table gives the statistics for a single scene. The first column represents a building extraction system. The next two columns give the percentage of building and background terrain correctly identified as such. The fourth and fifth columns show incorrect identification percentages for buildings and terrain. The next two columns give the breakdown (in percentages) of incorrect pixels in terms of false positives and false negatives. The last column gives the building pixel branching factor.

We believe that the quantitative results generated by the new evaluation method accurately reflect the subjective visual quality of the set of building hypotheses, when taken as a relative measure. Further, the building pixel branching factor provides a rough estimate of the amount of noise generated in the fusion process. Judging by these measures, we note that the final results of the hypothesis fusion process significantly improve the detection of buildings in a scene. In all of the scenes, the detection percentage for the final fusion is greater than the same percentage for any of the individual extraction system hypotheses, although the building pixel branching factor also increases due to the accumulation of delineation errors from the various input hypotheses.

It is worth noting that the results for the DC36B scene (Table 4-3) are substantially worse than those of the other scenes. This is in large part due to the fact that the DC36B scene has a low dynamic range of intensities, and the component systems used for these fusion experiments are inherently intensity-based. The building pixel branching factors reflect the poor performance of the component systems; in GROUPER's case, over 3 pixels are incorrectly hypothesized as building pixels for every correct building pixel. The fusion process, however, improved the building detection percentage noticeably over the percentages of the component systems.

We also note that several difficulties are attributable to performance deficiencies in the systems producing the original building hypotheses. The shadow-based detection and evaluation systems, SHADE and SHAVE, both use a threshold to generate "shadow regions" in an image. This threshold is generated automatically by BABE, a line-corner based detection system. In some cases, the threshold is too low, and the resulting shadow regions are incomplete, which results in fewer hypothesized buildings.

GROUPER, the shadow-based hypothesis clustering system, clusters fragmented hypotheses by forming a region (based on shadow-building edges) in which building structure is expected

to occur. This region is typically larger than the true building creating the shadow-building edge, and incorrect fragments sometimes fall within this region and are grouped with correct fragments. The resulting groups tend to be larger than the true buildings, and thus produce a fair number of false positive pixels.

SHAVE scores a set of hypotheses based on the extent to which they cast shadows, and then selects the top fifteen percent of these as "good" building hypotheses. In some cases, buildings whose scores fell in the top fifteen percent actually had relatively low absolute scores. This resulted in the inclusion of incorrect hypotheses in the final merger.

SHADE uses an imperfect sequence finder [1] to locate corners in the noisy shadow-building edges produced by thresholding. The sequence finder uses a threshold value to determine the amount of noise that will be ignored when searching for corners. In some situations, the true building corners are sufficiently small that the sequence finder regards them as noise, and as a result, the final building hypotheses can either be erroneous or incomplete.

4.4. Thresholding the accumulator image

As part of the scan-conversion fusion process, an accumulator image is produced that represents the "building density" of the scene. More precisely, each pixel in the image has a value, which is the number of hypotheses that overlapped the pixel. Pixels with higher values represent areas of the image that have higher probability of being contained in a man-made structure. Theoretically, thresholding this image at higher values and then applying connected region extraction techniques would produce sets of hypotheses containing fewer false positives, and these hypotheses would only represent those areas that had a high probability of corresponding to structure in the scene.

To test this idea, the accumulator images for each of the six scenes were thresholded at values of 2, 3, and 4, since four systems were used to produce the final hypothesis fusion. Connected region extraction techniques were then applied to these thresholded images to produce new hypothesis segmentations. The new evaluation method was then applied to these new hypotheses.

In each of the scenes, increasing the threshold from its default value of 1 to a value of 2 causes a reduction of roughly 20 percent in the number of correctly detected building pixels.

Evaluation results for the fusion process on DC37							
System	% Bld Detected	% Bkgd Detected	% Bld Missed	% Bkgd Missed	% False Pos.	% False Neg.	Br Factor
SHADE	37.5	98.2	62.5	1.8	15.0	85.0	0.294
SHAVE	47.2	96.8	52.8	3.2	26.8	73.2	0.408
GROUPER	48.7	95.8	51.3	4.2	32.6	67.4	0.508
BABE	58.9	97.2	41.1	2.8	28.5	71.5	0.278
FUSION	77.7	92.0	22.3	8.0	68.0	32.0	0.611
99 regions in ground truth							

Table 4-1: Evaluation statistics for DC37 hypothesis fusion



Figure 4-7: Monocular hypothesis fusion for DC36A

Evaluation results for the fusion process on DC36A							
System	% Bld Detected	% Bkgd Detected	% Bld Missed	% Bkgd Missed	% False Pos.	% False Neg.	Br Factor
SHADE	53.8	97.0	46.2	3.0	30.7	69.3	0.381
SHAVE	63.6	96.2	36.4	3.8	41.8	58.2	0.411
GROUPE	58.0	95.8	42.0	4.2	40.6	59.4	0.495
BABE	51.0	97.9	49.0	2.1	22.1	77.9	0.273
FUSION	80.9	91.9	19.1	8.1	74.3	25.7	0.682
51 regions in ground truth							

Table 4-2: Evaluation statistics for DC36A hypothesis fusion

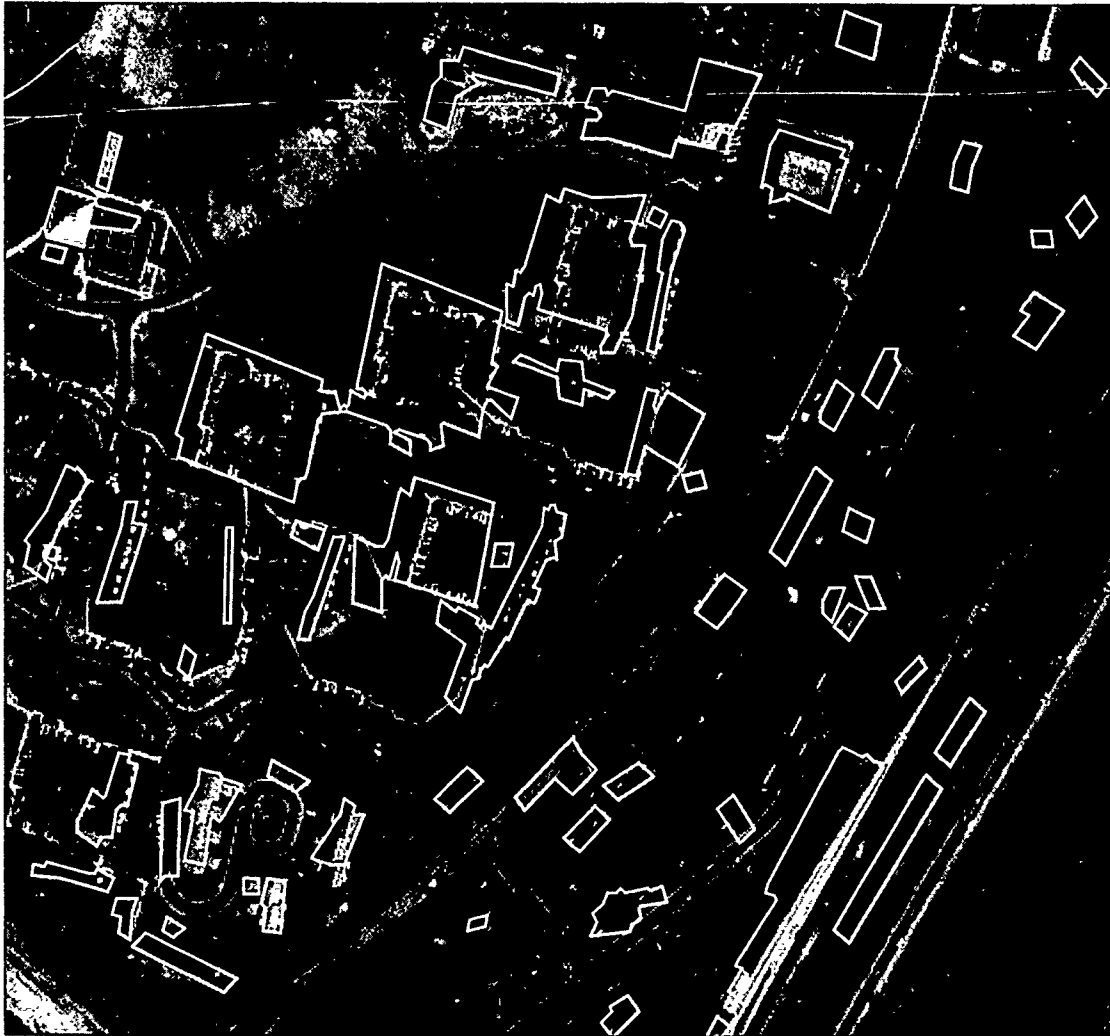


Figure 4-8: Monocular hypothesis fusion for DC36B

Evaluation results for the fusion process on DC36B							
System	% Bld Detected	% Bkgd Detected	% Bld Missed	% Bkgd Missed	% False Pos.	% False Neg.	Br Factor
SHADE	29.8	93.8	70.2	6.2	46.3	53.7	2.034
SHAVE	28.4	96.7	71.6	3.3	31.3	69.7	1.146
GROUPE	10.3	96.8	89.7	3.2	25.9	74.1	3.027
BABE	9.9	98.8	90.1	1.2	11.3	88.7	1.159
FUSION	49.8	89.2	50.2	10.8	67.8	32.2	2.126
133 regions in ground truth							

Table 4-3: Evaluation statistics for DC36B hypothesis fusion

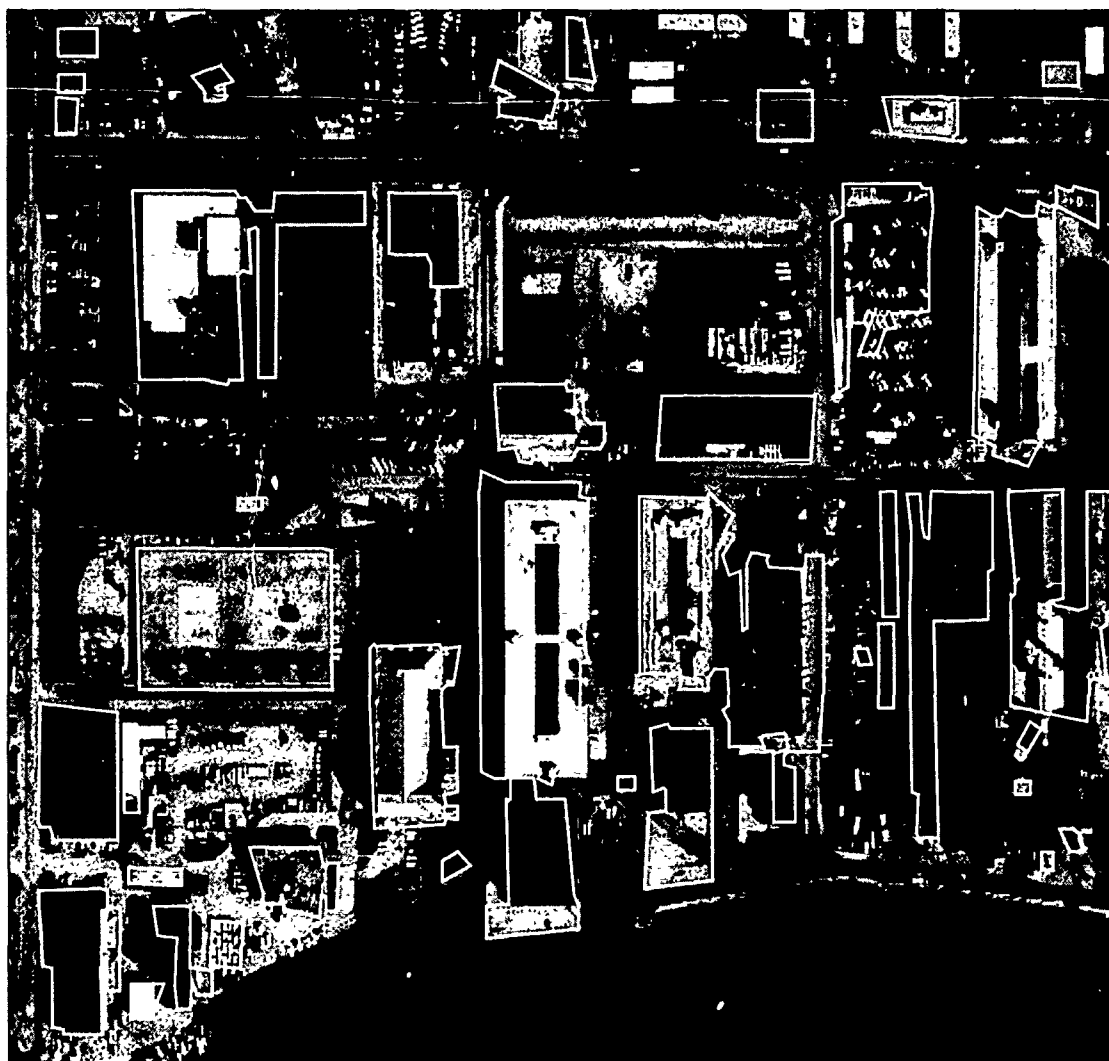


Figure 4-9: Monocular hypothesis fusion for DC38

Evaluation results for the fusion process on DC38							
System	% Bld Detected	% Bkgd Detected	% Bld Missed	% Bkgd Missed	% False Pos.	% False Neg.	Br Factor
SHADE	51.3	97.4	48.7	2.6	13.2	86.8	0.144
SHAVE	43.1	95.3	56.9	4.7	19.1	80.9	0.311
GROUPE	54.6	95.8	45.4	4.2	21.0	79.0	0.221
BABE	44.7	96.0	55.3	4.0	17.3	82.7	0.260
FUSION	74.7	90.6	25.3	9.4	51.5	48.5	0.360
53 regions in ground truth							

Table 4-4: Evaluation statistics for DC38 hypothesis fusion

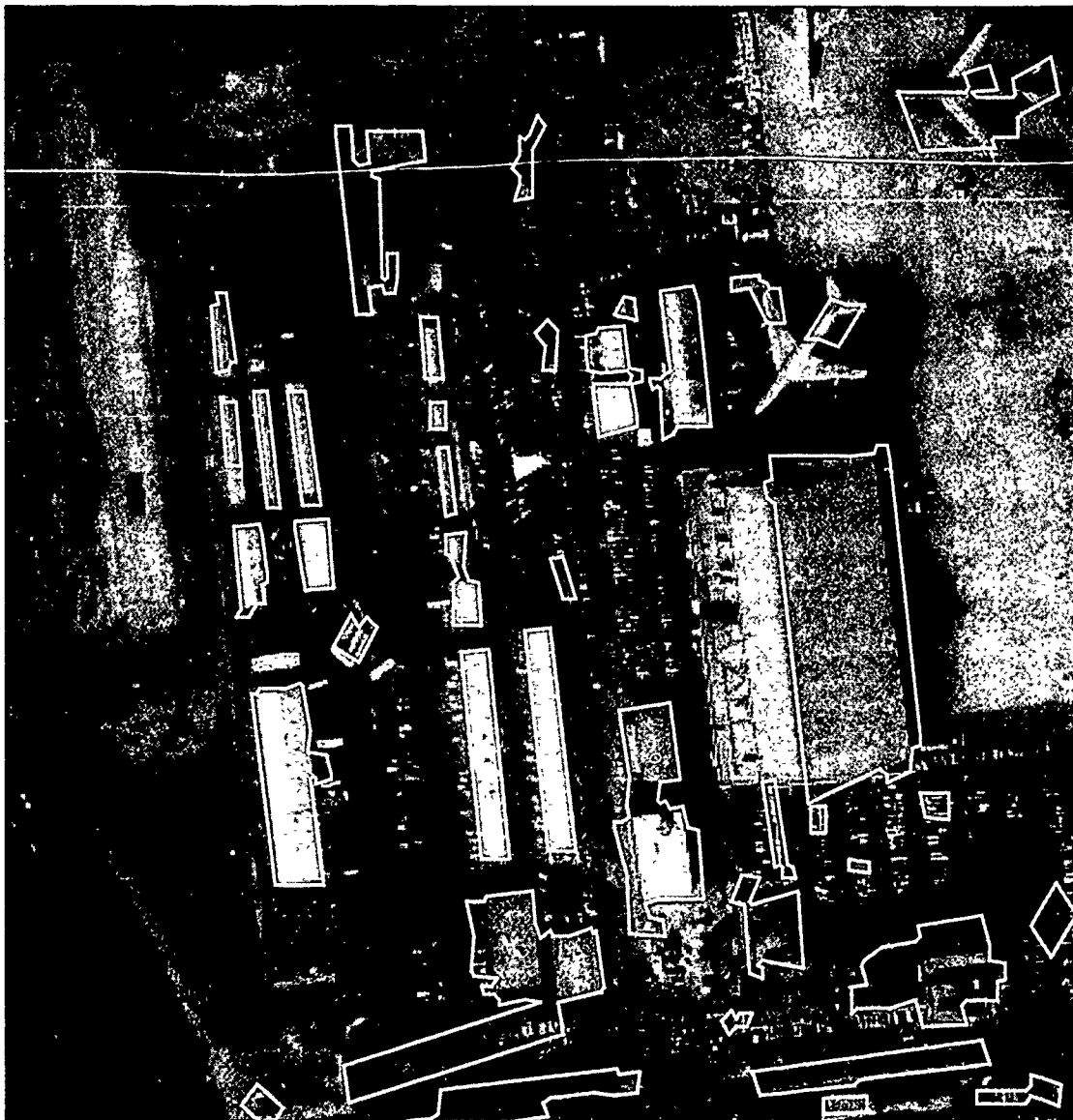


Figure 4-10: Monocular hypothesis fusion for LAX

Evaluation results for the fusion process on LAX							
System	% Bld Detected	% Bkgd Detected	% Bld Missed	% Bkgd Missed	% False Pos.	% False Neg.	Br Factor
SHADE	34.4	99.0	65.6	1.0	10.1	89.9	0.213
SHAVE	54.1	94.9	45.9	5.1	43.6	56.4	0.655
GROUPE	46.0	98.5	54.0	1.5	16.5	83.5	0.232
BABE	63.3	98.8	36.7	1.2	18.3	81.7	0.130
FUSION	73.0	92.9	27.0	7.1	65.0	35.0	0.687
26 regions in ground truth							

Table 4-5: Evaluation statistics for LAX hypothesis fusion

This suggests that a fair number of hypothesized building pixels are unique, i.e., several pixels can only be correctly identified as building pixels by one of the detection methods. Another interesting observation is that the building pixel branching factor roughly doubles every time the threshold is decremented. These observations suggest that thresholding alone may eliminate unique information produced by the individual detection systems, and that more work will need to be done to limit the number of false positives (and erroneous delineations) produced by each system, and by the final fusion as a whole.

4.5. Generating three-dimensional representations

The goal of three-dimensional scene analysis is to generate an interpretation of the imagery that is as close as possible to the actual scene under consideration. It is our belief that no individual computer vision technique can reliably provide a complete scene reconstruction. To achieve this goal, we will need to utilize multiple sources of information (which may be incomplete or inconsistent) and integrate them into a consistent interpretation of the scene. The method described in this paper integrates one type of monocular information: building delineations.

There are other types of information that can be integrated with these fused building delineations to allow the formation of three-dimensional representations. Since we have qualitative building boundary information, we can generate three-dimensional views with the integration of height information. This height information can be obtained from several visual cues as well, among these are shadow information and disparity information from the analysis of stereo imagery.

Figure 4-11 shows a perspective view for the DC37 scene, generated by the use of ground-truth terrain elevation values and building height segmentations. It is an accurate three

dimensional view of the scene structure using manual feature extraction techniques. Figure 4-12 shows a similar perspective view generated without manual height estimates for the terrain. Figure 4-13 shows a perspective view with structural height estimates automatically derived from a disparity map. The disparity map was generated by the fusion of disparity estimates produced by two stereo matchers, one area-based and one feature-based [5]. It is worth noting that height estimates of this nature do not constitute three-dimensional representations of the scene; a true representation would include building delineations, a transportation network of roads, and a digital elevation model. The information fusion approach provides a means for integrating image cues to produce the components of a true three-dimensional representation of the scene. In this particular case, the fusion of building boundaries (which are themselves fusions of building hypotheses) with disparity maps provides one component of the three-dimensional representation, qualitatively accurate building delineations and heights. In that sense, Figure 4-13 should be compared with the perspective view in Figure 4-12, since we do not utilize a terrain model in the fusion techniques described here.

Figure 4-14 shows another perspective view for the DC37 scene, with structural estimates derived from SHAVE by analysis of the lengths of the cast shadows of buildings [7]. SHAVE detects and delineates the shadows cast by each of the fusion building regions by walking from the shadow/building edge along the sun direction vector. At each pixel along the shadow/building edge an estimate of the shadow length is computed. The median length of the set of shadow vectors is computed for each building; this becomes the building shadow length estimate. Using the trigonometric relationship between building height, sun inclination angle, and length of the east shadow we can estimate the building height with good accuracy. In fact, this procedure is used regularly in manual photo interpretation. It is interesting to note that this view was

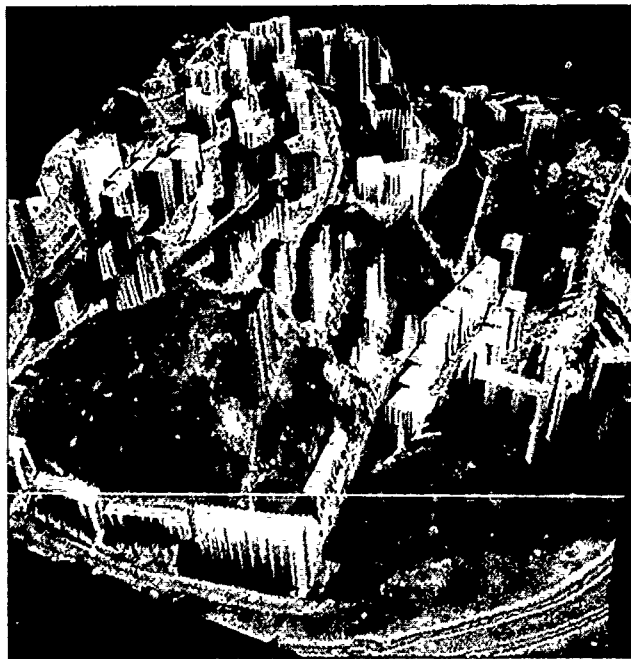


Figure 4-11: Perspective view for DC37 using ground-truth building and height data

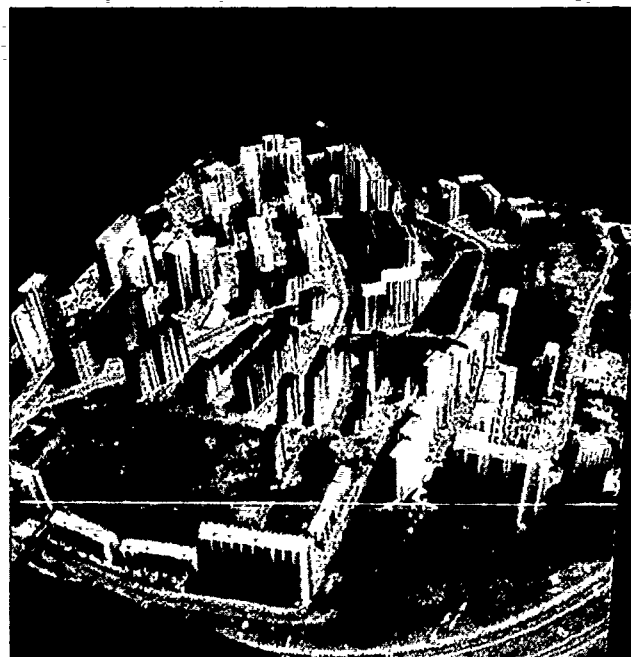


Figure 4-12: Perspective view for DC37 using ground truth building data only

generated solely from monocular analysis, no stereo information was utilized. Although stereo information is necessary in many situations for accurate height estimation, monocular analysis is capable of providing reasonable qualitative building delineations and heights.

5. Conclusions

This paper has described a simple method for fusing sets of monocular building hypotheses for aerial imagery. Scan-conversion and connected region extraction techniques were applied to produce mergers of sets of building hypotheses, and the results were analyzed by the use of an evaluation technique based on pixel coverage.

The simple hypothesis fusion approach developed here appears promising; the detection rate can be improved significantly by applying it to the results of several building detection systems. Much work remains to be done, however. Analysis of the fusion results has revealed shortcomings in each of the building detection systems, and there are also a

number of directions to pursue in terms of improving the intermediate and final fusions generated during the overall fusion process.

1. GROUPE is effective in clustering the fragmented hypotheses that are typically produced by BABE, but several of the grouped fragments do not correspond to building structure in the scene. Experimentation with disparity maps to refine these clusters is currently underway.

2. SHAVE's scoring system is simplistic and sometimes allows hypotheses with low shadow scores to pass as good hypotheses. Alternative scoring schemes might be explored.

3. SHADE's shadow segmentation and corner finding system can be improved. Work is currently underway on a method for iteratively approximating the location of corners in noisy

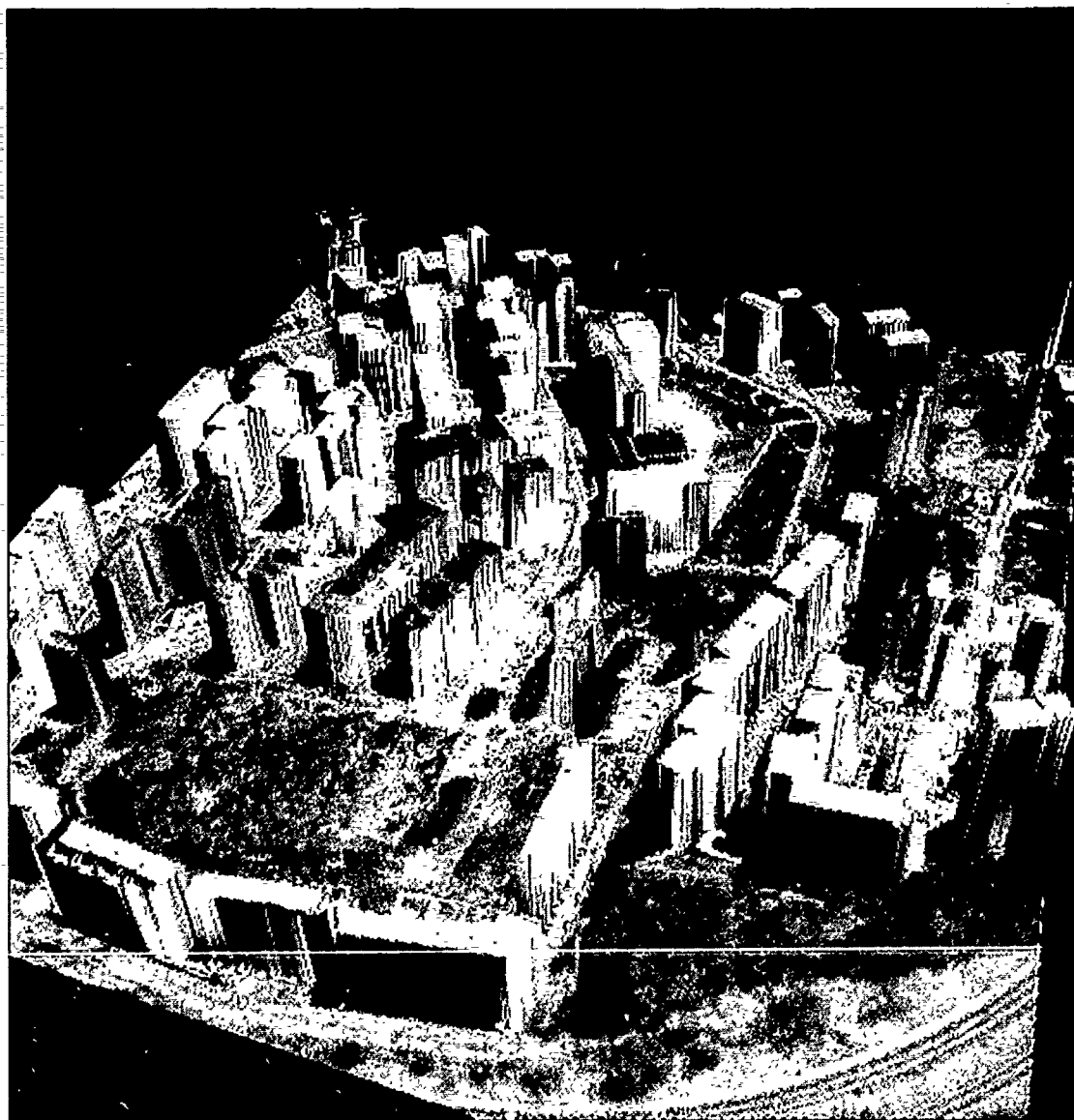


Figure 4-13: Perspective view for DC37 using stereo disparity information

lines by using an imperfect sequence finder to break lines at potential corners, and applying a gradient-based line evaluation function to score the breaks.

4. The fusion steps in the overall fusion process tend to increase the number of false positive pixels, and thresholding alone may not improve this without decreasing the number of correctly hypothesized pixels as well. The use of a refined disparity map, as well as the use of the original intensity image, may aid in eliminating false positive pixels from hypothesized regions in the final fusion. Alternatively, active contour models [8, 4] might be used to refine segmentations, using the fusion segmentations (possibly thresholded) as the initial seed to the process. This may prove difficult, however; fairly accurate estimates of the building boundaries will be necessary, and there may be

difficulties in recovering from local energy minima in complex high-resolution scenes.

5. Another interesting application of this fusion technique would be on binocular imagery. One could imagine merging hypotheses from the left and right images of a stereo pair to obtain an improved interpretation of a scene, since it is likely that the left and right hypothesis sets would differ due to changes in image perspective. Experiments are underway in this area.

A more general question concerns the effectiveness of simple fusion approaches such as the one described here. Certainly, one can envision other approaches for combining building hypotheses that would make use of *a priori* information about the systems producing the hypotheses to produce meaningful fusions of the individual hypotheses. It is unclear, however, whether such approaches would ultimately benefit from the additional complexity required to take advantage of such

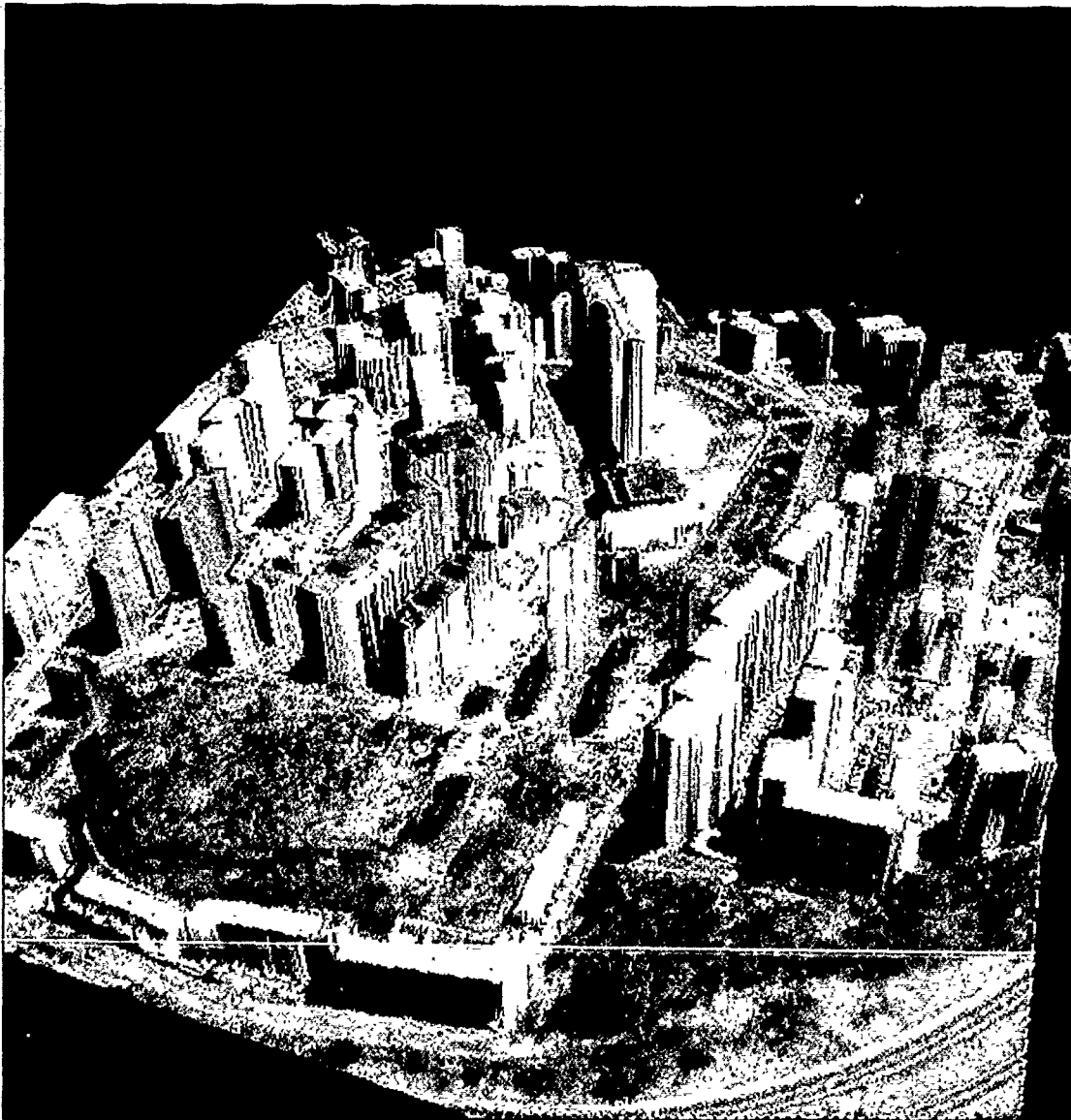


Figure 4-14: Perspective view for DC37 using monocular shadow analysis

knowledge. Although the results at this stage are rough, the fusion method developed here appears to be a simple and effective means for increasing the building detection rate for a scene, and may eventually provide a means for incorporating several sources of photometric information into a single interpretation of the scene.

6. Acknowledgments

We would like to thank the members of the Digital Mapping Laboratory for providing an interesting and congenial working environment. Particular thanks go to Yuan Hsieh and Frederic Perlant for interesting discussions about information fusion and building extraction techniques.

References

- [1] Aviad, Z.
Locating Corners in Noisy Curves by Delineating Imperfect Sequences.
Technical Report CMU-CS-88-199, Carnegie-Mellon University, December, 1988.
- [2] Aviad, Z., McKeown, D. M., Hsieh, Y.
The Generation of Building Hypotheses From Monocular Views.
Technical Report, Carnegie-Mellon University, 1990. to appear.
- [3] Fua, P., Hanson, A. J.
Resegmentation Using Generic Shape: Locating General Cultural Objects.
Technical Report, Artificial Intelligence Center, SRI International, May, 1986.
- [4] Fua, P., Hanson, A. J.
Objective Functions for Feature Discrimination: Theory.
In *Proceedings: DARPA Image Understanding Workshop*, pages 443-460. May, 1989.
- [5] Hsieh, Y., Perlant, F., and McKeown, D. M.
Recovering 3D Information from Complex Aerial Imagery.
In *Proceedings: 10th International Conference on Pattern Recognition, Atlantic City, New Jersey*, pages 136-146. June, 1990.
- [6] Huertas, A. and Nevatia, R.
Detecting Buildings in Aerial Images.
Computer Vision, Graphics, and Image Processing 41:131-152, April, 1988.
- [7] R. B. Irvin and D. M. McKeown.
Methods for exploiting the relationship between buildings and their shadows in aerial imagery.
IEEE Transactions on Systems, Man and Cybernetics 19(6):1564-1575, November, 1989.
- [8] Kass, M., Witkin, A., and Terzopoulos, D.
Snakes: Active Contour Models.
International Journal of Computer Vision 1(4):321-331, 1987.
- [9] Mohan, R., Nevatia, R.
Using Perceptual Organization to Extract 3-D Structures.
IEEE Transactions of Pattern Analysis and Machine Intelligence 11(11):1121-1139, November, 1989.
- [10] Nicolin, B., and Gabler, R.
A Knowledge-Based System for the Analysis of Aerial Images.
IEEE Transactions on Geoscience and Remote Sensing GE-25(3):317-329, May, 1987.

Map-Based Localization: The "Drop-Off" Problem

William B. Thompson, Herbert L. Pick, Jr.,
Bonnie H. Bennett, Marian R. Heinrichs,
Steven L. Savitt, and Kip Smith
University of Minnesota
Minneapolis, MN 55455

Abstract

Navigation based on maps requires frequent solutions to the *localization* problem. Localization is the process of establishing a match between particular locations in the environment and the corresponding locations on a map. Most often, localization involves determining the viewpoint and thus the location and heading of the navigating agent on the map. The solution requires both low-level extraction of image and map features and high-level problem solving to establish likely correspondences while avoiding prohibitively expensive search. We present a formalism within which the localization problem can be studied, information about how expert human map users deal with localization, and aspects of a preliminary computational model of the process.

1 Introduction

Localization is the process of establishing a match between particular locations in the environment and the corresponding locations on a map. Commonly, the environment location of interest is the viewpoint and viewing direction (i.e., the "where am I?" problem). Figures 1

and 2 illustrate a typical localization problem. Figure 1 shows a view of Moran Canyon in Grand Teton National Park. (Though we are primarily interested in ground level imagery, this particular example was taken from a helicopter flying approximately 1,300 meters above Jackson Lake.) Figure 2 shows a section of a topographic map which includes both the viewpoint for the picture and much of the terrain visible in the picture. The localization task involves determining the viewing position and direction on the map which corresponds to what is seen in Figure 1. In Figure 2, the true viewpoint location and direction is marked by a \rightarrow .

At an abstract level, localization can be modeled as three interacting processes (Figure 3). Two perceptual processes identify appropriate map and image structures, a third process actually establishes correspondence. Perception needs to operate in both a top-down and bottom-up manner. Operating bottom-up, perceptual components of the process return the location and type of prominent features. Operating top-down, they search the data for features of a particular type at a particular location. In the third process, features which are candidates for matching are found in one set of features and then are searched for in the other set. The matching is bi-directional; that is, map properties can be searched for among image features or image features can be searched for among map features. The search is guided by a priori



Figure 1: View of Moran Canyon.

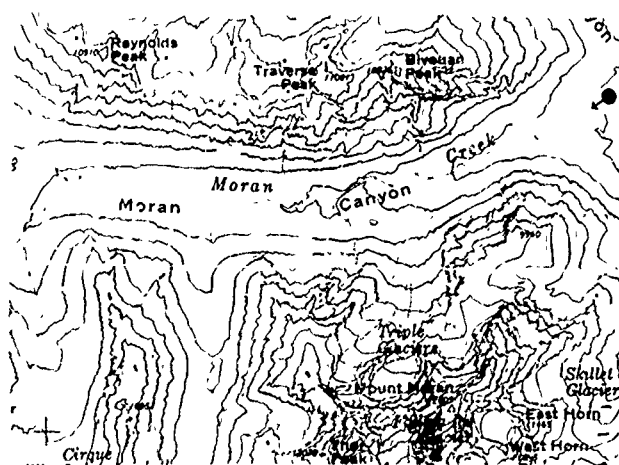


Figure 2: Topographic map of Moran Canyon.

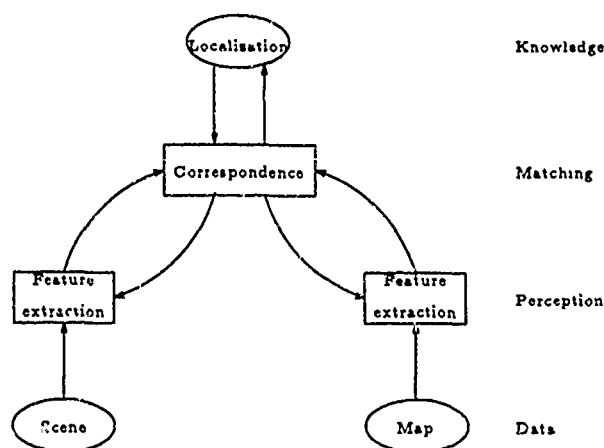


Figure 3: Top-level model of localization process.

knowledge of the likely viewpoint, together with heuristics that reduce the potential complexity. The localization problem itself is solved when correspondence is established between the observation point and a map location. Much of our research is aimed at understanding what features and feature properties are relevant to the perception level and what strategies are used at the matching level to guide the search.

Automated solutions to the localization problem are of obvious utility for mobile robotics in large-scale, outdoor terrain. In addition, a more precise understanding of the processes involved in localization can aid human map users through better training procedures. Finally, outdoor localization provides a challenging research environment within which to advance image understanding technology. Most of the "shape-from-X" techniques that have been developed are ineffective in large-scale, outdoor terrain due to the long distances involved and the complex reflectance models that prevail. New low-level analysis techniques based on occlusion cues and properties such as aerial perspective will be required.

In this paper, we describe a preliminary computational model for solving one type of localization problem. In addition we outline relevant information learned from studies we have done involving expert map users solving a variety of realistic outdoor and laboratory tasks. Subsequent reports will elaborate this model for a broader class of localization problems and show how lower-level vision modules and high-level spatial reasoning need to interact in order to perform localization while navigating outdoors.

2 Approach to the Problem

Localization problems can be characterized in terms of how much a priori information is available about likely observations points (Figure 4). At one end of this continuum, *drop-off* problems involve substantial initial uncertainty in viewing location and/or direction. (The name comes from the extreme case in which an observer is "dropped off" into a totally unfamiliar environment.) In *updating* problems, the task is to maintain a sense of the current position with respect to a map as the current

position changes incrementally due to locomotion. We have initially focused our research on drop-off problems, since many of the techniques for solving drop-off problems are likely to be part of the solution to updating problems. In addition, the drop-off problem gives us a sense of base-level performance for map-based localization under a high degree of uncertainty.

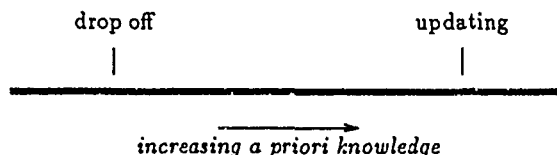


Figure 4: Variations in a priori knowledge affect the nature of the localization process.

Almost all of the previous work on localization using vision has been directed at updating problems. Knowledge of expected position (typically from dead reckoning) is used to predict visual features which are then searched for in the image. Deviations between expected and observed images are used to update the estimate of current location. While updating plays a necessary role in outdoor navigation, it is not sufficient in and of itself to solve the localization problem. Over the long distances and time intervals involved in large-scale outdoor navigation, dead reckoning errors accumulate to prohibitively large values, the maintenance of a visual fix on features necessary for updating becomes increasingly difficult, and dealing with the occlusion and disocclusion of tracked features introduces special problems.

Over shorter time intervals, it may be possible to start with an initial solution to the localization problem, use this to visually identify significant image features and note the corresponding map features, use low-level visual correspondence methods to track these features when moving, then use triangulation techniques to solve for the new current location. Even if this is possible, a continuous 360° view of the scene must be available and substantial computational resources are required.¹ Outdoor environments often have areas in which no distinctive features are visible. In such situations, it is essential that a method be available for reacquiring a sense of location on the map after moving into more varied terrain. Furthermore, low-level visual tracking of topographic features is not as simple as it might at first appear. The irregular shape of most topography together with the frequency of curving slopes presents significant problems. Relatively small movements of the observation point can produce significant changes in appearance of a single feature. Even worse, visually prominent aspects of one topographic feature may smoothly move to another feature as the viewpoint is changed. (E.g., a visual high point may correspond to a particular hill in

¹ Consider a real application: During tank battles, inertial position sensors drift at least one nautical mile per hour, global positioning system (GPS) information may be unavailable or unreliable, and it is clearly not possible for the tank crew to continuously and precisely keep track of all visual changes in the local topography.

one view and a different hill in a subsequent nearby view, without any obvious event in the imagery signaling that a different hill has come into view.) Finally, the frequent occlusion and disocclusion of structures needed for triangulation requires the visual acquisition of new features, presenting an additional possibility for significant error.

Real world topography involves complex shapes at many different scales. Even with an accurate map, the number of characteristic views (nodes in the aspect graph) grows rapidly with increasing uncertainty in viewing location. As a result, the combinatorics of the drop-off problem are such that it is usually not possible to use a verification strategy in which an expected view is matched against actual imagery. Instead, localization becomes more like a recognition problem in which the task is to decide what region of the map can act as a "model" to adequately explain visible portions of the scene.

While people can do object recognition rapidly and with little apparent effort, they have considerably more difficulty with localization problems. Effective utilization of a topographic map appears to combine use of visual skills with substantial problem solving. Localization is a high-level perceptual activity quite different from the recognition tasks that are more commonly studied. This suggests that localization may be an application in which lower-level image understanding techniques and methods from artificial intelligence may be naturally combined. It also suggests that the development of computational solutions for the localization problem can benefit significantly from research on how expert map users solve similar problems.²

3 Relationship Between Localization and Recognition

Vision is a process that extracts information about *what* and *where* from an image. Most of the research on higher-level vision has concentrated on recognition tasks. In recognition, the fundamental problem is to identify *what* is in the image. Aspects of the problem involving shape and position (*where*) may be both necessary and difficult, but they are typically subsidiary to the identification process. In contrast, issues of *where* are central to localization.

Many of the computational tools that have proven useful for recognition turn out to be also relevant to localization. Use of such formalisms allows a more formal specification of the localization problem while at the same time highlighting similarities and differences with existing recognition algorithms.

Grimson separates the problem of recognition into three conceptual components: selection of appropriate

subsets of image features to match against object models, selection of appropriate object models, and establishment of correspondences between model and image features [Grimson, 1990]. Much research has focused solving the correspondence problem using *pose estimation* or *alignment* techniques in which the correspondence between model and image features is coupled with the estimation of the transformation between model and image coordinate systems (e.g., [Huttenlocher and Ullman, 1987]). Localization involves these same conceptual components, though there are distinct and significant differences.

In outdoor navigation, the relevant "model" is a representation of the topographic features visible from a particular vantage point. Because the number of vantage points is effectively unbounded, we no longer have a set of discrete models. Rather, the needed model of the topography must be assembled adaptively from the map. Severe combinatorial problems will result if this assembly of map features is not carefully constrained.

The selection of appropriate image features for matching is rather more straightforward, since all topographically distinctive visible features are potentially relevant. (In recognition, the image is typically cluttered with a large number of features unrelated to the object to be identified. For localization, the "clutter" is in the models, not in the image.) Proficient map users exploit this fact by driving the generation of hypothesized viewpoints based more on features visible in their view of the scene than on a search through possibly relevant map features. Still, the number of visible scene features usually presents combinatorial difficulties. Success in localization seems to involve organizing these features into easily matchable *configurations*.

Correspondence requires a one-to-one matching between particular subsets of map (model) and image features. Grimson describes this as a constraint satisfaction problem, distinguishing between unary constraints which apply to single pairings of an image feature with a model feature and n-ary constraints which apply to larger sets of pairings. (Grimson actually considers nothing more complex than binary constraints.) In localization, unary constraints consist of equivalent identifications of map and image features (e.g., "hill"), possibly combined with descriptive information about the feature (e.g., "high"). N-ary constraints relate configurations of basic features (e.g., "two hills separated by a saddle").

In object recognition, pose estimation involves the determination of the transformation that will best match a particular model to a given set of image features. For three-dimensional models and two-dimensional image features, this transformation typically involves up to six degrees of freedom: two of translation, one of scale (or equivalently, depth), and three of rotation. Finally, the projection of the transformed model onto the image plane must be determined.

The situation is rather more complex for localization in outdoor environments. Recognition is not based on generic, three-dimensional models. Instead, topography leads to $2\frac{1}{2}$ -D models, since the environment can be thought of as a 2-D, horizontal surface that has been

²The fact that localization seems to be harder for people than object recognition does not necessarily argue against studying human performance in order to build computational models. Experience with expert systems suggests that it is *easier* to build these programs based on how people solve difficult problems than based on seemingly effortless "common sense", since the processes used to solve the more difficult problems are easier to access experimentally.

distorted out of the plane. A map is in effect a 2-D, downward-looking view of this $2\frac{1}{2}$ -D surface. The images on which localization must be based are horizontal-looking views of the same surface. Thus, in matching model (map) to image, we always have a 90° rotation to deal with. This *perspective shift* between downward-looking and horizontal-looking views is quite distinct from the other translations and rotations of the map necessary to establish the viewpoint.

The 90° perspective shift between map and image has important implications for the sorts of lower-level image understanding techniques necessary to support localization. Knowing that the shift occurs constrains, to some extent, the problem of finding the complete transformation which specifies the solution to the localization problem. Unfortunately, the "on end" view of the topographic model together with the difficulty of accurately determining range over long distances using passive vision means that it is not possible to extract a precise quantitative geometric description of the scene from the available images and then match this against the map.

4 Strategies for Localization

Expert map users use six distinct processes in solving localization problems. Competence in all six seems required for effective performance. It is likely that these same procedures will be required in automated systems which solve localization problems without precise *a priori* information on viewing position.

4.1 Reconnaissance

The purpose of reconnaissance is to gather information prior to the creation and/or evaluation of specific hypotheses about the viewing position. Perceptually distinctive topographic properties of the map or scene that are potentially relevant to establishing map-image correspondences are identified.³ Reconnaissance involves an examination of either map or image features in isolation. The most successful map users seem to spend most of their reconnaissance time examining image features, organizing the information from the environment into a cohesive representation of features and configurations. Initial reconnaissance focusing on the map seems less successful.

Localization problem solving is almost always initiated by an extended period of reconnaissance. The search is conducted broadly, without any particular focus except as to distinctiveness and relevance of features. Follow-up episodes of reconnaissance generate additional information and can be prompted by three different situations. Acquisition of additional information is common during the evaluation of a hypothesis. The additional information is required whenever the current information is

insufficient to establish the hypothesis. Follow-up reconnaissance is also useful during the refinement of a hypothesis that is being accepted. The additional information typically serves to fine-tune the hypothesis. The most common use of follow-up reconnaissance is as a "strategic regrouping" after the rejection of a hypothesis. This regrouping appears to serve the same purpose as the initial extended reconnaissance, the gathering of information required to support the targeting of a new hypothesis.

4.2 Map Orientation

Map orientation involves relating the direction and scale of the map to the visible scene. If an accurate compass is not available, the map is aligned with the general lay of the land. An approximate calibration is established between the scene and the map contour interval and distance scale. Map orientation can occur at a variety of points in the problem solving process. It typically is required only once, unless hypotheses based on a previously determined value are proving hard to verify.

4.3 Feature Matching

The major activity during the localization task is matching features in the image to features in the map or vice versa. Feature matching does *not* require the existence of a hypothesis about viewing location. Such matching can establish possible general correspondences between the image and the map, facilitating the generation of specific hypotheses. Once hypotheses have been established, feature matching plays a key role in evaluation.

Feature matching is based on a common identification and a similar characterization of topographic structures in the map and in the image. Identification is done in terms of a set of labels and properties that is often specific to a particular geologic landform. In the rolling terrain of southeastern Minnesota, the most common features attended to are hills and valleys. Matching for the presence or absence of an individual hill or valley is not particularly diagnostic of location. Accordingly, map users more commonly attend to properties of these features rather than just the existence of the feature. To differentiate among similar hills and valleys, they focus on relative size, elevation, and gradient (steepness).

Most map users tend to impose a bipolar classification system to differentiate properties of features. Features are either large or small, narrow or broad, steep or shallow. Comparison is another common strategy to differentiate features. One feature is said to be larger, broader, or steeper than another.

4.4 Configuration Matching

Configuration matching serves the same purposes as feature matching. The only difference between the two are that the pieces of information that are being attended to are assemblies of features. Configurations are specified in terms of the features of which they are composed and the relationships between those features. These relationships include purely topological descriptions (e.g., behind, in front of, next to), ordinal relations (e.g., taller than), and quantitative properties (e.g., actual eleva-

³Criteria for distinctiveness and potential relevance can vary significantly over different landforms. The kinds of features relevant to localization in Minnesota are very different than those relevant to the glaciated topography of the mountains in the western United States. Anecdotal evidence suggests that even expert map users may require adaptation before effectively dealing with novel sorts of terrain.

tion). Expert map users tend to do more configuration matching and less feature matching than do less proficient individuals. The complexity of the configurations is usually relatively small, however, typically involving two to four individual features. Competence in map reading appears to depend on the accurate establishment of appropriate configurations for matching.

Configurations constrain the matching process more effectively than do individual features. There are fewer matches to "a hill with a dip and a ridge" than there are to individual hills, to individual small valleys, and to individual ridges. By bundling features together into configurations, the map user effectively restricts search to models with more unique descriptors.

Experienced map users appear to follow a pair of simple but highly effective heuristics as they assemble configurations of features in the image. The first heuristic restricts configurations to features that are contiguous. Features that are joined together to form configurations are invariably physically adjacent (e.g., "the flat area that slopes down and then up again to a ridge"), rather than just adjacent in the imagery due to occlusion. Map users in the field have often been observed to trace out in the air with a finger the connection between features as they construct a configuration.

The second, less-rigorously applied heuristic, restricts configurations to features that align along a line-of-sight. The majority of configurations (perhaps 80%) used by map users are composed of contiguous features that fall along or parallel to a line-of-sight, along an azimuth that extends away from the viewer. Most of the remaining configurations (the other 20%) focus on the distribution of features along prominent ridge-lines that cut across the viewing angle. The common characteristic of these assemblages is their linearity. Whenever a feature in a configuration does not line up, explicit reference is made to its non-linearity (e.g., the crook in a ridge-line or the slight offset in a string of hills and valleys).

Most configurations are assembled in accord with both heuristics. Both derive their power from the fact that they disallow configurations that could be products of accidental viewpoints. Both connectivity and linearity are viewpoint invariant properties of the image that survive the transformations required for matching. ([Lowe, 1987] emphasizes a similar importance for viewpoint invariant configurations of features in object recognition.)

4.5 Hypothesis Generation and Evaluation

A hypothesis posits a distinct map location and direction as corresponding to the viewing position. The hypothesis is initially triggered by the possible map-image correspondence between a small number of features or correspondences. Hypothesis evaluation proceeds by examining other image and map features or configurations using expectations about correspondences derived from the hypothesis. Often, a brief reconnaissance of a local region in the map and/or image will be required to identify additional features and configurations useful in the evaluation process. The strategies involved have much in common with those used in other diagnostic tasks (e.g., see [Johnson *et al.*, 1988]).

While viewpoint invariance is desirable in the spatial arrangements of features that define a configuration, *viewpoint dependence* is obviously necessary for hypotheses. A hypothesis must necessarily describe the relationship of topographic features to the viewpoint. Our experience with expert map users suggests that they use rather simple, qualitative descriptions for these relationships rather than a more sophisticated trigonometric analysis. Whether this is the best approach to the problem or only a consequence of the difficulty people have in making complex quantitative judgements is not yet clear.

The search through alternate hypotheses can proceed in a variety of ways. A breadth-first strategy, typically not very effective, generates a large number of hypotheses before attempting to evaluate any of them. The generation of each individual hypothesis is based on a small number of features – often only one. More focused search strategies generate successively more precise hypotheses based on increasingly richer sets of configurations. These focused searches may alternate generation and evaluation or may generate a small set of possibilities and then simultaneously examine all at once.

The most common error made by map users is the failure to generate the correct answer as one of the candidates in a set of initial hypotheses. This type of error seems to have as its source an inadequate reconnaissance of the scene in the map user's immediate vicinity. An overly simple description of the location (e.g., "I'm on a big hill" or "This ridge is steep") ends up matching the most prominent "big hill" or "steep ridge" in the map, without concern for the greater constraints that would be provided by a richer set of configurations.

A second type of error is made during the evaluation of a hypothesis. A common evaluation strategy is to examine the map for features or configurations that can be expected in the image if the hypothesized location were correct. If the model of the environment generated from the map is poorly constructed, it is all too easy to "explain away" expectations that are not realized. The source of this type of error is the failure to use the model to identify disconfirmatory evidence in the image. This is an instance of confirmatory bias, a common source of failure in human problem solving [Wason, 1960, Mynatt *et al.*, 1977]. The chance for error is enhanced in the localization task by the inherent imprecision of the model upon which the evaluation is made. This is one situation in which we might expect automated perceptual systems to perform better than their human counterparts.

4.6 Conclude

Hypothesis evaluation leads to the tentative rejection or confirmation of hypotheses that have been generated. A final step in the localization process produces the best estimate of actual location and viewing direction. Depending on the search strategy used, this may be based on a comparison of the likelihoods of competing hypotheses or may simply be the identification of a single hypothesis which survived a sequential generate-and-test procedure.

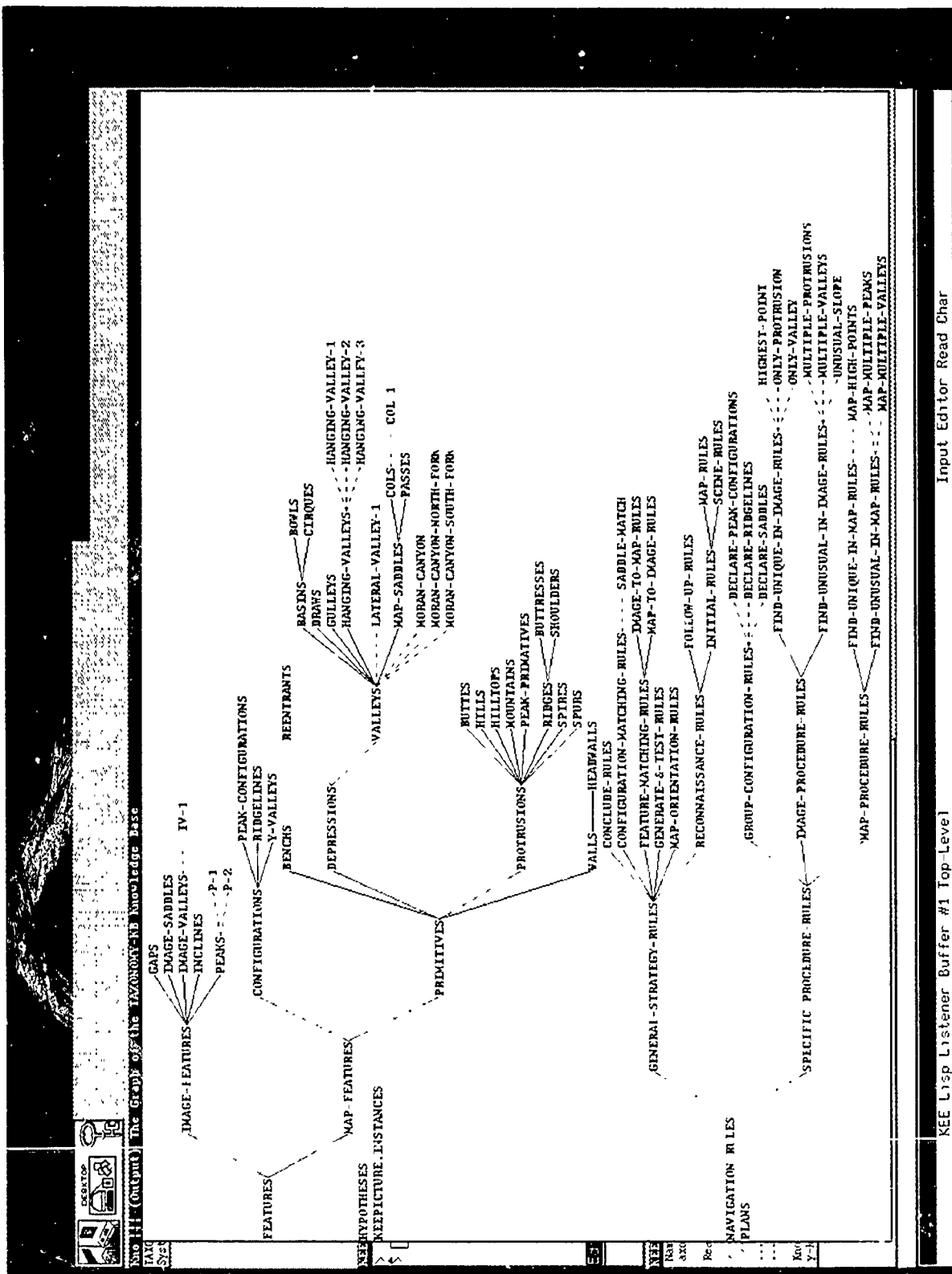


Figure 5: Matching of map and image features.

5 A Computational Architecture for Localization

We have completed the preliminary specification of a computational architecture for the problem solving aspects of drop-off problems. The model includes a *taxonomy knowledge base* for aiding in the recognition of topographic features and the assembly of configurations, *image* and *map knowledge bases* for representing information specific to the problem at hand, and a *hypothesis knowledge base* for posting information on currently active hypotheses about viewpoint or scene-image correspondences. A set of procedures forms a *control structure* for recognizing features, assembling configurations, and posting, evaluating, refining, and accepting or rejecting hypotheses. In addition, the control structures have access to lower-level components responsible for extracting primitive features from map and imagery.

Figure 5 shows an example of map data partially instantiated against partially interpreted image data. The taxonomy knowledge base is used to create a hierarchy starting with topographic features and continuing on down through the solid (subclass) links to map and image features, primitives and configurations, etc. In this example, the image knowledge base consists of the frames representing two peaks (P-1 and P-2) and a valley (IV-1) which have been recognized in the image. These frames have been attached appropriately into the taxonomy domain by membership (dashed) links. The map knowledge base consists of three hanging valleys (hanging-valley-1, -2, and -3), three canyons (moran-canyon along with its -south-fork and -north-fork), and a col (col-1). These are attached to appropriate places in the taxonomy hierarchy via more membership links.

Several of the control structure procedures are shown in Figure 5. These procedures are divided into two classes. General strategy rules include reconnaissance (both initial and follow-up), map orientation, feature matching (both image to map, and map to image), configuration matching, hypotheses generation and evaluation, and conclusions. Specific procedures perform tasks such as grouping configurations and attentional processes such as looking for unique or unusual data like prominent high points or unusual configurations.

6 Lower-level Image and Map Understanding

Extraction of map features is aided greatly by the availability of accurate DTM (digital terrain model) data, since the interpretation of contour data involves a number of subtle interpolation problems. If DTM data is available, extraction of features such as peaks, ridges, and valleys can be done using relatively straightforward mathematical operators [Shapiro *et al.*, 1988]. A significant recognition problem remains, however, since important distinctions exist between features in the same class (e.g., a cirque is a very different feature than a canyon, though both are instances of valley features). This is different from the classic object recognition task, since

terrain classification depends on sometimes subtle shape properties, not on a geometrically precise object model.

The problem of assembling configurations is difficult not only because the criteria for choosing members of the configuration is seldom clear, but also because there is no obvious way in which to determine the spatial relationships within a configuration. This problem arises because the individual features have spatial extent, thus limiting the degree to which relationships such as "adjacent to" can be effectively utilized. The fact that expert map users organize configurations in a linear structure may be caused, in part, by the need for finding a compact representation of spatial organization within the configuration.

The extraction of image features also suffers from the lack of precise object models. In addition, the primitive structures needed for feature identification are not well defined. As with other image understanding situations, a large number of effects can generate the same or similar patterns on an image. Simple edge detection is clearly not enough as a basis for finding topographically relevant image features. Many open questions remain in this aspect of our research.

The extraction of image features based on edges requires that only edges likely to be due to topographic effects be identified. Two approaches seem promising. One, similar to methods used in other recognition applications, involves organizing local edge elements into larger segments likely to correspond to some meaningful scene structure. (See [Sha'ashua and Ullman, 1988] and [Mohan and Nevatia, 1988] for examples in the domain of object recognition.) The second involves understanding the specific constraints that exist on image edges generated by topographic structures.

Figure 6 provides one example of using information about topography to generate constraints on edges. The figure shows a sketch of a ridge viewed from slightly different directions. In the right view, we see the ridge in profile. In the left view, the ridge is seen more end-on and the faces on both sides of the ridge have become visible. If the topography consists of approximately planar faces, then a ridge can be characterized in terms of its *rise angle* (the angle of the ridge itself relative to horizontal) and the *break angles* of each of the faces (the slope of the face measured along its fall line). For a horizontal viewing direction, the projection process is such that the angle of the ridge as projected into the image is never less than the rise angle. Furthermore, the projected ridge angle in the image for ridges seen in

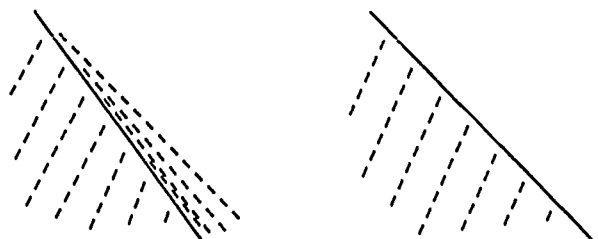


Figure 6: Ridge line seen from different vantage points.

profile is never more than the break angle of the hidden face. Thus, knowledge of the minimum rise angles and maximum break angles that are common in the scene constrain to an interval the projected ridge lines in the image. In most realistic situations, the viewing angle is sufficiently close to horizontal for this effect to be useful. Even in extremely rugged terrain, break angles are seldom more than 45° , thus providing a useful way to evaluate edges in the image.

Figures 7–11 illustrate a number of the lower-level image and map understanding problems that arise in localization. Figure 7 shows topographic features extracted from the DTM data using methods described in [Shapiro *et al.*, 1988]. Black lines indicate ridges, white lines indicate valleys. The features are overlaid onto an elevation image in which lighter values indicate higher altitudes. (Only a portion of the map shown in Figure 2 is shown.) Figure 8 shows the output of an appropriately thresholded Canny edge detector applied to the image in Figure 1. Figure 9 shows the Canny edges which have passed a multi-stage filtering operation involving spatial coincidence across scales, minimum edge length, and expectations about edge orientations. Figures 10 and 11 illustrate how simple textural patterns can aid in the identification of topographic structures. Figure 10 shows image edges filtered to preserve only those oriented down and to the right. Figure 11 shows edges oriented down and to the left. Concentrations of edges in Figure 10 indicate rightward facing slopes. Concentrations of edges in Figure 11 indicate leftward facing slopes. (The smaller clusters of edges in the upper left of Figure 11 are faces associated with the far walls of side valleys branching off from the main canyon.)

7 Related Work

Our research draws on a diverse range of past work. Localization is a fundamental problem in mobile robot navigation, and several different types of solutions have been developed to address it. Approaches for automated interpretation of reconnaissance imagery relate directly to the problems of determining locations in large-scale space. Finally, an extensive literature exists on human competence in map reading.

7.1 Computational approaches

Solutions to the localization problem in mobile robot navigation take two forms. Both approaches match the actual image with the scene that is expected given an estimated location, but differ in the level at which the matching takes place. In the first approach, a 3-D model of the scene and an estimate of the viewing location is used to predict what the 2-D image should look like. Edges from the predicted image are compared with edges found in the actual image. One example of this approach uses map data to project a potential image given a downward-looking perspective from an estimated position [Ernst and Flinchbaugh, 1989]. This potential image is then run through a low-level matcher which compares it to the actual incoming image. The resulting correspondences are then used to refine the estimated position. Another example, the PSEIKI sys-

tem, also uses an estimated position which is derived from motion information to generate a two-dimensional projection of the structure in the expected scene [Andress and Kak, 1988]. Correspondences are found by using the Dempster-Shafer formalism. The HILARE system, too, uses motion information to estimate position, and explicitly represents positional uncertainty numerically [Chatila and Laumond, 1985]. The part of the world model near the estimated position which best corresponds to what is currently being perceived is then found using a global matching approach.

The second approach to localization in mobile robot navigation matches the expected scene with the actual image using landmarks at the level of objects and places. Distinguishable objects in the environment are identified using perceptual systems. The bearing and range to each landmark is then used to orient the system with respect to a "world model" (i.e., map) of the environment. One example of this approach is the NX robot, which, during an exploration phase, determines locally distinctive places by finding sensory features which are maximized at that place [Kuipers and Byun, 1987]. This *signature* is then used during later navigation to recognize the place. Levitt *et al.* developed a model of landmark-based localization in which landmarks are used in a highly error tolerant manner to partition the environment into places which are recognized by the landmarks configurations seen there [Levitt *et al.*, 1987, Levitt *et al.*, 1988]. Another method addresses the localization problem by combining low-level tracking or visual "servoing" with high-level perceptual verification using *milestones*. These milestones are defined in terms of landmarks such as buildings, for example, and their bearing [Arkin *et al.*, 1987, Fennema *et al.*, 1988]. Another system generates a 2-D and partial 3-D scene model from the observed scene. The matching problem is then solved by using object groupings and spatial reasoning [Nasr *et al.*, 1987].

Conventional approaches to landmark-based localization require that the identification and global position of landmarks be known a priori with a high degree of precision, and that perceptual systems exist which can accurately identify these landmarks and precisely determine their relative position with respect to the robot vehicle. Object recognition that is at the same time both general and robust is difficult to achieve. As a result, errors in landmark recognition will be common. In many environments, precisely localized landmarks may be scarce. Finally, the ambiguity associated with landmark-based navigation can lead to a combinatorial explosion of cases that must be analyzed. If there are many landmarks of the same type, then the complexity of the task matching landmarks to map features grows quickly.

The integration of sensed data with maps is central to many navigation tasks. Map-to-image matching has been extensively studied within the context of reconnaissance imagery (e.g., [Nevatia and Price, 1982, Clark, 1983, McKeown and Denlinger, 1984, Hwang, 1984, McKeown *et al.*, 1985]). Typically, meaningful features are found in the image and then matched to corresponding map features. Common matching items in-



Figure 7: Extracted topographic features.

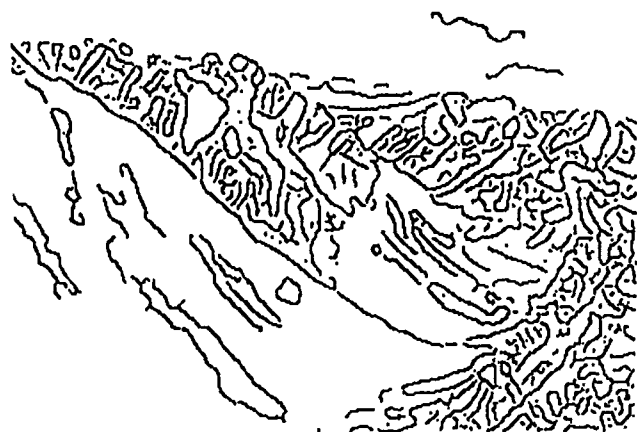


Figure 8: Canny edges from Fig. 1.



Figure 9: Filtered edges.



Figure 10: Diagonal edges from Fig. 1.



Figure 11: Edges oriented in opposite diagonal.

clude cultural features such as roads, cities, and airports, along with terrain features such as rivers, coastlines, and so on. [Little, 1982] describes one of the few map-to-image systems that makes heavy use of *topographic* features. Ridge lines are found in a digital terrain model and placed in correspondence with brightness discontinuities in an image. The matching is aided by information about illumination angle which is used to predict which ridges in the elevation model will generate distinctive changes in brightness. In all of these cases, imagery and maps have had a common, "downward-looking" perspective where both imagery and maps have a similar, two-dimensional coordinate system. The correspondence problem is essentially one of 2-D registration.

In the problems we are considering here, imagery has a near "horizontal-looking" perspective which is qualitatively different from the downward view common to nearly all maps. There has been relatively little work relating horizontal-looking imagery with maps. The work closest to our own is that of Lavin who was interested in a problem complementary to that of map matching [Lavin, 1979]. He investigated the creation of topographic maps from sketches of occlusion boundaries. Only a very simple model of topography involving uniform Gaussian shaped hills was used. Thus, many of the complexities encountered in more realistic situations were avoided. Related to both Lavin's work and the methods for matching reconnaissance imagery and maps are techniques for automatically rendering terrain views based on both aerial photography and elevation data [Quam, 1985]. Appropriate coordinate transformations and resampling are done to produce a horizontal-looking view from the original downward-looking photograph.

The perspective shift associated with combining visual data with other representations such as maps is related to several other three-dimensional reconstruction problems. Koenderink developed a relationship between the 3-D structure of solid objects and the topology of projected contours [Koenderink, 1984]. Giblin and Weiss describe how surface descriptions can be recovered from projected contours [Giblin and Weiss, 1987]. Neither of these approaches, however, is directly applicable to our problem. Complex terrain cannot be modeled as a simple, solid object. Furthermore, the inaccuracies of lower-level image analysis algorithms is likely to defeat any method based on the topology of projected contours. Finally, Shepard's work on mental rotations may provide some insight into human performance in perspective shift tasks [Shepard and Metzler, 1971].

7.2 The psychology of using maps

An extensive literature in psychology and cartography deals with problems associated with reading and using maps and the associated problem of recognizing aspects of scene geometry relevant to localization. While little of this literature deals with the actual processes involved in localization, it does provide useful insight into the sorts of computational models likely to be effective. Knowledge about the performance of expert map users can aid in understanding the heuristic strategies necessary to establishing correspondences between a map and an image.

Lower-level image understanding methods which utilize passive vision are unlikely to work much better than humans. As a result, information about the limitations of human vision in large-scale, outdoor environments is potentially of great relevance in developing computational solutions for vision-based localization.

Preliminary research of ours using both protocol analysis of observers thinking aloud as they solved a drop-off localization problem and a memory paradigm of observers recalling photographic images suggested that much attention during observation of natural scenes may be devoted to qualitative topographic features [Heinrichs *et al.*, 1989]. Certainly there was more mention of such features than precise metric characteristics. Among the kinds of features noted were a variety of convex features (hills, ridges, rises), concave features (valleys, sinks, holes, etc.), inclinations (level plateaus and slopes). Although the organization of spatial knowledge has mainly been studied in urban or restricted laboratory environments the indications are that features or landmarks exert a strong influence on one's use of spatial information. For example, [Sadalla *et al.*, 1980] demonstrated that certain salient features serve as reference points for organizing spatial information. Once established, these reference points have a privileged role in spatial orientation, with one result being that the subjective distance between reference points and non-reference points is not symmetrical.

Other research has shown that spatial information is hierarchically organized. This is evidenced by the fact that making judgments (or thinking about) particular locations will facilitate subsequent independent judgments about locations that are physically nearby [Hirtle and Jonides, 1985, McNamara, 1986]. Another factor which contributes to such hierarchical organization is the extent to which various physical factors compartmentalize a space [Kosslyn *et al.*, 1974]. Distances between locations within the same subspace will often be judged as smaller than equivalent distances between locations in different subspaces. These subspaces might be defined by physical barriers such as rivers or fences, by optical barriers such as the edge of a field, or by political boundaries such as state or city lines.

Analysis of individual differences in map reading performance has also been used as a way of investigating the processes of extracting information from maps. [Chang *et al.*, 1985] studied how eye movements during reading topographic maps were related to individual differences in map reading experience. They found that the eye fixations of experienced map readers were shorter and more often focused on task relevant areas than those of inexperienced readers. Sholl and Egeth [Sholl and Egeth, 1982], in a systematic psychometric approach, related performance on a number of topographic map performance tasks to several more general standard psychometric measures. The map tasks such as land form identification, slope identification, spot elevation, and terrain visualization were factor analyzed, yielding two major factors, one described as a spatial visualization factor and the other an altitude estimation factor. Surprisingly, standard tests of spatial ability are not highly

related to the spatial visualization map reading factor whereas verbal-analytic measures are. A standardized measure of mathematical ability is related to the altitude estimation factor, yet finding the altitude of points on a topographic map or finding the highest and lowest elevations wouldn't seem to involve very sophisticated mathematics. The authors suggest that the relationship is due to the arithmetic aspect of mathematical ability. In general, the results seem to suggest that our standardized tests don't reflect very well the abilities used in a practical skill like topographic map reading.

An obvious approach to understanding the processes underlying extraction of information from topographic maps is the use of information processing paradigms. There are few such studies, but one example [Eley, 1988] has examined the effect of differences in orientation in view point on speed of matching a map position to the topography of a surface. Subjects were shown a segment of a topographic map for inspection. After they had a chance to study the map, a point and direction of view was indicated on the map perimeter. Their task was then to imagine what the land surface would look from that perspective. When they were satisfied that they knew how the surface would look they pressed a button which presented a representational drawing of a surface. They then had to indicate whether the surface drawing corresponded or not to the specified view. Of particular interest was how the time required to imagine the view from the specified orientation was related to the viewing direction. Typical mental rotation results were obtained. The greater the required viewing direction deviated from the subject's own orientation the longer the reaction time to press the button for the drawing. In a second experiment reaction time was measured for land surface views at different elevations. Results indicated that an elevation providing a viewpoint of 30 degrees above horizontal was more effective than either higher or lower elevations. The effects on map reading performance of the mismatch in orientation between map and environment has also been found with street maps [Levine *et al.*, 1984].

Although space perception has been a topic of study for over one hundred and fifty years only so-called depth perception, the perception of the radial distance of objects from the observer, has received systematic intense investigation [Haber, 1985]. Psychophysical research has been concerned with how observers are able to obtain information about a 3-D world from 2-D sensory input. The few studies conducted in rich outdoor environments have suggested that a linear relationship exists between perceived and physical distance for spaces relevant to navigation. Unfortunately, all of these studies were done in flat open fields. No such studies have been carried out on even sloping or irregular (not to mention cluttered) landscapes.

Laboratory studies of the perception of the slant of surfaces indicate reasonable sensitivity to relative inclination as specified by optical texture and linear perspective (e.g., [Flock, 1965]). However, there is only one report of observation of the slope of a natural incline and that suggested that frontally viewed slopes were seen as

steeper than they really were [Smith and Smith, 1965]. This result is consistent with anecdotal reports of hills often appearing steeper than they actually are when one is traversing them by foot or in a vehicle. In work preliminary to the present project, slopes were estimated from photographs at points for which the actual slopes varied from about 3 to 25 degrees. Results indicated a linear relationship between actual and perceived slope. Consistent with the observation by Smith and Smith slopes were perceived as steeper than they actually were.

The limited research that exists on reading of topographic maps is interesting and tantalizing. The results suggest a rather sophisticated skill, but neither an analysis of individual differences nor of task processes provides an adequate understanding of the nature of that skill. One reason is simply that there is relatively little research. Another is that the tasks used are artificial in two respects. The materials used are not realistic. The samples of maps themselves are real but often only very small segments are used. When the experimental tasks involve relating maps to the environment, the environment is typically represented by relatively impoverished sketches which may, on the one hand, emphasize features that wouldn't be as clear with natural terrain or, on the other hand, omit the incredible richness of natural terrain. The tasks are also artificial in the problems posed. Subjects may be asked only to find a high or low spot, to judge the qualitative nature of a land form, etc., and they are usually not even asked to solve a localization problem.

8 Implications For Training

A better understanding of the formal nature of the localization problem and the processes likely to be successful in solving localization problems has the potential for improving the training of map users. Knowledge about the perceptual limitations leading to localization errors can be used to warn map users of potential difficulties. Search and evaluation strategies which reduce the combinatorics and minimize ambiguity can be taught, while strategies known to be less effective can be avoided.

Map reading problems take a variety of forms. Localization tasks such as updating and drop-off problems involve map-image correspondence. Some other tasks focus solely on maps. These would include route planning, determination of intervisibility ("when looking from point A to C, would intermediate point B be visible?"), finding highest and lowest station points in an area, determining the direction of water flow, etc. Accuracy and efficiency in reading maps is important for both kinds of problems and accuracy and efficiency in perception of the scene is a necessary prerequisite for the correspondence problem. In addition, solving the map-image correspondence problem requires use of a variety of information processing and problem solving strategies. Establishing such a correspondence involves relating a two-dimensional plan perspective with an encoded third dimension to an eye-level view of a three-dimensional environment.

How accurate is our perception? As noted in section 7.2, the perception and memory of scene and map in-

formation is subject to a variety of distortions. Recall the evidence that slope of inclines is over-estimated and that distances between locations in different subspaces are over-estimated. Heights of hills and mountains can also be misperceived. Erroneous judgments of the relative heights of distant and nearer peaks may be caused by not properly taking into account one's own altitude and misperceiving whether one's own direction of gaze is above or below eye level. Such an error may have been a factor in a military plane crash [Haber, 1987].

Similar distortions occur in processing of map information (e.g., [Tversky, 1981, Tversky and Schiano, 1989]). For example, people tend to remember map features as more aligned than is in fact the case. In one case Tversky demonstrated that people will remember continents such as North and South America as more aligned with the cardinal axes of maps than they actually are. Thus, South America is considered to be almost directly south of North America. Such distortions can account for further erroneous judgments such as New York being typically judged as east of Santiago while in fact it is west. Similar distortions occur with more local features, such as city streets. In addition, features that are diagonal tend to be rotated toward cardinal frame axes and are remembered more nearly parallel or perpendicular to major features.

Where do problems arise in the process of solving map-image correspondence problems? On the basis of background literature and our prior work done related to this project, it has been possible to identify some problematic aspects of the solution process. Recall the studies mentioned above that indicate misalignment between map and scene increase the difficulty of the map reading problem. Orienteers are trained in always aligning their map to the scene as they traverse a course. They have found that this increases the efficiency of their map following when time is a premium and helps to reduce errors. It would be easy to demonstrate to the trainees the effects of misalignment between map and scene.

In our initial empirical work on map reading, protocols were collected from persons solving drop-off localization problems. Analysis of these protocols suggests that for drop-off problems a successful strategy is to work from the visible scene to the map. Apparently, specifying the scene features and configurations of features constrains the areas on the map that need to be examined. When this strategy is not successful, one reason is that the local features around the station point are misperceived. Trainees should be alerted to this danger. We observed a number of problematic strategies. One of the most frequent was a "garden path" kind of error in which attention was focused on one or on a very few possible solutions. Incorrect hypotheses were pursued over a long chain and disconfirming evidence was discarded or explained away.

In general trainees can be apprised of both successful strategies and procedures that are likely to lead to trouble. Trainees can be drilled on such problems and their errors pointed out. Unfortunately, field problems are very time consuming. Simulated problems in the classroom are a possibility [Barsam and Simutis, 1984].

However, the simulations need to be developed carefully. In one attempt to develop a laboratory analog to the actual drop-off problem using photographic images we found that the simulation distorted the process by eliminating some of the early stages of problem solving.

9 Discussion

Localization, particularly localization involving drop-off problems, fits well into the conceptual formalism that has been used for several successful approaches to object recognition. The most significant difference is that for localization, predefined object models are not available. Instead, drop-off problems require that models of the scene be created from information supplied on maps. This is possible only after preliminary hypotheses about viewing position and direction have been generated. (Updating problems are easier, in part, because the task of assembling models is much more straightforward.) The lack of predefined object models introduces significant added complexity over that involved in object recognition. This complexity can be overcome by the use of heuristic search strategies which combine sophisticated problem solving with more traditional perceptual processing.

Our formalism predicts the desirability of focusing the search based on an initial reconnaissance of the image before any exploration of the map occurs. This strategy is in fact often observed in expert map users. An interesting contrast occurs with localization problems involving a rapidly moving observer. Before the availability of more sophisticated navigation aids, fighter pilots were trained to do localization by first checking a stopwatch to determine the time spent on the current leg of the flight plan, then estimating their current location on the map and looking for distinctive map features, and finally attempting to visually locate those features in the environment [Ullman, 1990]. In our terminology, this corresponds to an initial reconnaissance focusing on the map - a sensible strategy when elapse time provides an initial guess as to position and the imagery is changing at a substantial rate.

As with alignment methods for object recognition, localization involves the recognition of viewpoint invariant configurations of features. Tentative correspondences between such configurations in map and image data can be established prior to the generation of hypotheses about the viewpoint defined transformation between map and image.

Future work will concentrate on strategies for additional types of localization problems and low-level computer vision requirements for localization. Segmentation algorithms tuned to outdoor scenes are required, as are techniques for recognizing topographic features such as peaks, ridges, and valleys in an image. The ability to actively move the view point will be explored, since an active observer can better determine scene properties such as slopes, while at the same time moving to distinctive positions that aid in the generation of viewpoint hypotheses.

Acknowledgments

Elizabeth Stuck provided important criticism on earlier drafts of this paper. Ting-Chuen Pong contributed to the detection of topographic features from DTM data. Some of the results are based on work by Daniel Montello and Catherine Sullivan.

References

- [Andress and Kak, 1988] K. Andress and A. Kak. Evidence accumulation and flow of control. *AI Magazine*, 9(2):75-94, 1988.
- [Arkin et al., 1987] R. C. Arkin, E. M. Riseman, and A. R. Hanson. AuRA: An architecture for vision-based robot navigation. In *Proc. DARPA Image Understanding Workshop*, pages 417-431, Los Altos, CA, February 1987. Morgan Kaufmann.
- [Barsam and Simutis, 1984] H. F. Barsam and Z. M. Simutis. Computer-based graphics for terrain visualization training. *Human Factors*, 26:659-665, 1984.
- [Chang et al., 1985] K. T. Chang, J. Antes, and T. Lenzen. The effects of experience on reading topographic relief information: Analyses of performance and eye movements. *The Cartographic Journal*, 22:88-94, 1985.
- [Chatila and Laumond, 1985] R. Chatila and J.-P. Laumond. Position referencing and consistent world modeling for mobile robots. *1985 IEEE International Conference on Robotics and Automation*, pages 138-145, March 1985.
- [Clark, 1983] J. Clark. Integration of imagery and cartographic data through a common map base. *Proc. SPIE*, 359:163-168, 1983.
- [Eley, 1988] M. G. Eley. Determining the shapes of land surfaces from topographical maps. *Ergonomics*, 31:355-376, 1988.
- [Ernst and Flinchbaugh, 1989] M. D. Ernst and B. E. Flinchbaugh. Image/map correspondence using curve matching. In *AAAI Symposium on Robot Navigation*, pages 15-18, March 1989.
- [Fennema et al., 1988] C. L. Fennema, Jr., E. M. Riseman, and A. R. Hanson. Planning with perceptual milestones to control uncertainty in robot navigation. In *Proceedings of the SPIE*, 1988.
- [Flock, 1965] H. R. Flock. Optical texture and linear perspective as stimuli for slant perception. *Psychological Review*, 72:505-514, 1965.
- [Giblin and Weiss, 1987] P. Giblin and R. Weiss. Reconstruction of surfaces from profiles. In *First Int. Conf. on Computer Vision*, pages 136-144, Washington, DC, June 1987. Computer Society Press of the IEEE.
- [Grimson, 1990] W. E. L. Grimson. *Object Recognition by Computer: The Role of Geometric Constraints*. The MIT Press, 1990.
- [Haber, 1985] R. N. Haber. Toward a theory of the perceived spatial layout of scenes. *Computer Vision, Graphics, and Image Processing*, 31:282-321, 1985.
- [Haber, 1987] R. N. Haber. Why low-flying fighter planes crash: Perceptual and attentional factors in collision with the ground. *Human Factors*, 29:519-532, 1987.
- [Heinrichs et al., 1989] M. R. Heinrichs, D. R. Montello, C. M. Nusslé, and K. Smith. Localization with topographic maps. In *Proceedings of the AAAI Symposium on Robot Navigation*, pages 29-32, March 1989.
- [Hirtle and Jonides, 1985] S. C. Hirtle and J. Jonides. Evidence of hierarchies in cognitive maps. *Cognitive Psychology*, 13:208-217, 1985.
- [Huttenlocher and Ullman, 1987] D. P. Huttenlocher and S. Ullman. Object recognition using alignment. In *First Int. Conf. on Computer Vision*, pages 102-111, Washington, DC, June 1987. Computer Society Press of the IEEE.
- [Hwang, 1984] S. S. V. Hwang. *Evidence accumulation for spatial reasoning in aerial image understanding*. PhD thesis, University of Maryland, December 1984.
- [Johnson et al., 1988] P. E. Johnson, J. B. Moen, and W. B. Thompson. Garden path errors in diagnostic reasoning. In L. Bloc and M. J. Coombs, editors, *Computer Expert Systems*, pages 395-427. Springer-Verlag, 1988.
- [Koenderink, 1984] J. J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13:321-330, 1984.
- [Kosslyn et al., 1974] S. M. Kosslyn, H. L. Pick, Jr., and G. R. Fariello. Cognitive maps in children and men. *Child Development*, 45:707-716, 1974.
- [Kuipers and Byun, 1987] B. Kuipers and Y. T. Byun. A qualitative approach to robot exploration and map learning. In *Proc. of the IEEE Workshop on Spatial Reasoning and Multi-Sensor Fusion*, pages 390-404, Los Altos, CA, 1987. Morgan Kaufmann.
- [Lavin, 1979] M. A. Lavin. Analysis of scenes from a moving viewpoint. In P. H. Winston and R. H. Brown, editors, *Artificial Intelligence: An MIT Perspective*, pages 17-82. MIT Press, 1979.
- [Levine et al., 1984] M. Levine, I. Marchon, and G. Hanley. The placement and misplacement of you-are-here maps. *Environment and Behavior*, 16(2):139-157, 1984.
- [Levitt et al., 1987] T. S. Levitt, D. T. Lawton, D. M. Chelberg, and P. C. Nelson. Qualitative navigation. In *Proc. DARPA Image Understanding Workshop*, pages 447-465, Los Altos, CA, February 1987. Morgan Kaufmann.
- [Levitt et al., 1988] T. S. Levitt, D. T. Lawton, D. M. Chelberg, K. V. Koitzsch, and J. W. Dye. Qualitative navigation II. In *Proc. DARPA Image Understanding Workshop*, pages 319-326, Los Altos, CA, April 1988. Morgan Kaufmann.
- [Little, 1982] J. J. Little. Automatic registration of Landsat MSS images to digital elevation models. *Proc. of IEEE Workshop on Computer Vision: Representation and Control*, pages 178-184, 1982.

- [Lowe, 1987] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355-395, 1987.
- [McKeown and Denlinger, 1984] D. M. McKeown, Jr. and J. L. Denlinger. Map-guided feature extraction from aerial imagery. *IEEE Workshop on Computer Vision: Representation and Control*, pages 205-213, 1984.
- [McKeown et al., 1985] D. M. McKeown, Jr., W. A. Harvey, Jr., and J. McDermott. Rule-based interpretation of aerial imagery. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-7:570-585, 1985.
- [McNamara, 1986] T. P. McNamara. Mental representations of spatial relations. *Cognitive Psychology*, 18:87-121, 1986.
- [Mohan and Nevatia, 1988] R. Mohan and R. Nevatia. Using perceptual organization to extract 3-d structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11:1121-1139, 1988.
- [Mynatt et al., 1977] C. R. Mynatt, M. E. Doherty, and R. D. Tweeny. Confirmation bias in a simulated research environment: An experimental study of scientific inference. *Quarterly Journal of Experimental Psychology*, 29:85-95, 1977.
- [Nasr et al., 1987] H. Nasr, B. Bhanu, and S. Schaffer. Guiding an autonomous land vehicle using knowledge-based landmark recognition. In *Proc. DARPA Image Understanding Workshop*, pages 432-439, Los Altos, CA, February 1987. Morgan Kaufmann.
- [Nevatia and Price, 1982] R. Nevatia and K. E. Price. Locating structures in aerial images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-4:476-484, 1982.
- [Quam, 1985] L. H. Quam. The Terrain-Calc system. In *Proc. DARPA Image Understanding Workshop*, pages 327-330, Los Altos, CA, 1985. Morgan Kaufmann.
- [Sadalla et al., 1980] E. K. Sadalla, W. J. Burroughs, and L. J. Staplin. Reference points in spatial cognition. *Journal of Experimental Psychology: Human Learning and Memory*, 6(5):516-528, 1980.
- [Sha'ashua and Ullman, 1988] A. Sha'ashua and S. Ullman. Structural saliency: The detection of globally salient structures using a locally connected network. In *Second Int. Conf. on Computer Vision*, pages 321-327, Washington, DC, December 1988. Computer Society Press of the IEEE.
- [Shapiro et al., 1988] L. G. Shapiro, R. M. Haralick, and T. C. Pong. The use of the facet model and the topographic primal sketch in image analysis. In C. Brown, editor, *Advances in Computer Vision*, volume II. Lawrence Erlbaum Associates, 1988.
- [Shepard and Metzler, 1971] R. N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171:701-703, 1971.
- [Sholl and Egeth, 1982] M. J. Sholl and H. E. Egeth. Cognitive correlates of map reading ability. *Intelligence*, 6:215-230, 1982.
- [Smith and Smith, 1965] O. W. Smith and P. C. Smith. An illusion of slant in nature. *Perceptual and Motor Skills*, 20:1108, 1965.
- [Tversky and Schiano, 1989] B. Tversky and D. J. Schiano. Perceptual and conceptual factors in distortions in memory for graphs and maps. *Journal of Experimental Psychology: General*, 118:387-398, 1989.
- [Tversky, 1981] B. Tversky. Distortions in memory for maps. *Cognitive Psychology*, 13:407-433, 1981.
- [Ullman, 1990] S. Ullman, 1990. Personal communication.
- [Wason, 1960] P. C. Wason. On the failure to eliminate hypotheses in a conceptual task. *Quarterly Journal of Experimental Psychology*, 12:129-140, 1960.

B-snakes: implementation and application to stereo*

Sylvie Menet[†], Philippe Saint-Marc[‡] and Gérard Medioni

Institute for Robotics and Intelligent Systems

Powell Hall 204

University of Southern California

Los Angeles, California 90089-0273

Abstract

We present B-snakes: a new implementation of snakes using parametric B-splines. This active contour model exhibits advantages of B-splines: compact representation, local control and the possibility to include corners. This implementation is significantly faster without loss of generality. Experiments on delineation of building roofs in stereo aerial images are also presented.

1 INTRODUCTION

Real-world images are often noisy and too complex to expect local, low level operations to perform a complete analysis. Higher level features have to be derived and used in order to get a better delineation of objects.

When there exist enough constraints, it is possible to use deformable models, which adapt to the data, an example being "snakes" [Kass *et al.*, 1988].

We present an implementation of such models based on parametric B-spline approximation, which offers many advantages. Among them, it provides a compact local representation of a curve, in terms of its control-points. Furthermore, B-splines have the ability to represent corners, that is, to locally override smoothness constraints. A new active contour model is built using this B-spline approximation for a curve and is called a "B-snake". These B-snakes converge much faster than snakes and can include corners without invoking specific models.

As an application, B-snakes are used to precisely outline the boundaries of building roofs in stereo pairs of urban scenes, given an initial rough outline from a standard stereo matching algorithm [Cochran and Medioni, 1989].

The paper is organized as follows: we briefly review snakes and their applications, then give details of our B-snake implementation, and illustrate the methodology on the accurate delineation of building tops in stereo pairs of urban scenes.

2 SNAKES

A snake is a deformable continuous curve, whose shape is controlled by internal forces (the implicit model) and external forces (the data). Internal forces act as a smoothness constraint, and external forces guide the active contour towards image features.

Let $v(s) = (x(s), y(s))$ be the parametric description of the snake ($s \in [0, 1]$). Its total energy can be written as:

$$\begin{aligned} E_{snake} &= \int_0^1 E_s(v(s)) ds \\ &= \int_0^1 [E_{int}(v(s)) + E_{ext}(v(s))] ds \end{aligned} \quad (1)$$

with:

$$E_{int}(s) = \frac{1}{2}(\alpha(s) |v_s(s)|^2 + \beta(s) |v_{ss}(s)|^2) \quad (2)$$

The goal is to find the snake that minimizes equation (1), given some external energy adapted to image features to extract ($E_{edge} = -|\nabla I(x, y)|^2$, for example) and internal energy whose expression is given by (2). The first order term makes the snake act like a membrane and the second order one like a thin plate. This energy is the regularizing term of the minimization.

The minimization of (1) is solved by using the calculus of variations and resolving Euler equations, and yields the following equations in the discrete case [Kass *et al.*, 1988]:

$$\begin{cases} Ax + F_x(x, y) = 0 \\ Ay + F_y(x, y) = 0 \end{cases} \quad (3)$$

where $F = E_{ext}$ depends on the image features to extract and A is a pentadiagonal matrix depending on α and β .

This system of equations in (x, y) is solved by introducing an energy dissipation functional to dissipate the kinetic energy during the motion. Let γ be the Euler step size. The expression of the snake as a function of time is then:

$$\begin{cases} x_{t+1} = (A + \gamma I)^{-1}(\gamma x_t - F_x(x_t, y_t)) \\ y_{t+1} = (A + \gamma I)^{-1}(\gamma y_t - F_y(x_t, y_t)) \end{cases} \quad (4)$$

*This research was supported in part by DARPA contract F33615-C-1436

[†]Supported by a grant from the French Direction Générale de l'Armement, contract ERE 89/1460/DRET/DS/SR. Address: ONERA, 29 Ave de la Div Leclerc, 92320 Chatillon/Bagneux, France.

[‡]Permanent address: Matra-SEP Image et Informatique, Signal and Image Processing Laboratory, BP 235 "Les Miroirs", 38 Bd Paul Cézanne, Guillaucourt, 78052 St-Quentin en Yvelines Cédex, FRANCE

$(A + \gamma I)^{-1}$ can be calculated by LU decompositions in $O(n)$ time (with n being the length of the snake).

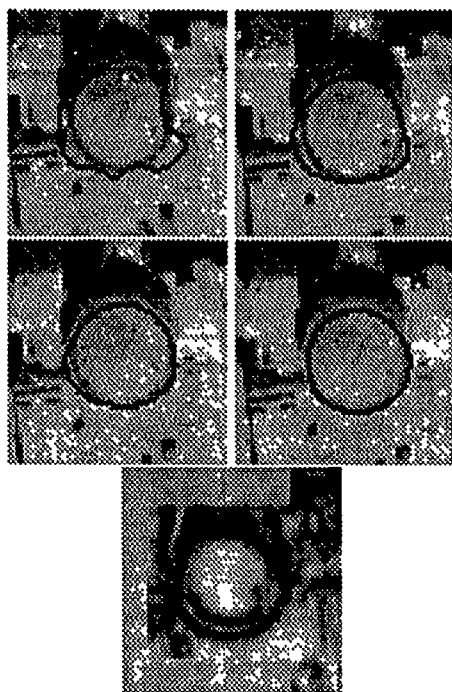


Figure 1: Example of snake convergence, the external energy is the negated gradient.

Figure 2 shows an example of convergence.

This active contour model fits in an interactive human-machine environment when the user supplies an initial estimate of the object to extract and the snake is used to refine the results [Kass *et al.*, 1988; Fua and Hanson, 1989b; Fua and Hanson, 1989a; Fua and Leclerc, 1988]. However, it is also useful in an automatic processes when a first estimate is given by a first processing level [Ferrie *et al.*, 1989; Zucker *et al.*, 1988].

This tool has been applied in motion [Kass *et al.*, 1988], in stereo matching [Kass *et al.*, 1988; Fua and Leclerc, 1988], and, more generally, it can be used to match a deformable model to an image by means of energy minimization.

Different implementations have been performed, for example, Fua [Fua and Hanson, 1989a] uses a tool from information theory: he minimizes an objective function that is the length of encoding the result. This methodology is general and applies to object recognition using generic models. Amini [Amini *et al.*, 1988] uses dynamic programming to minimize the energy, and can handle hard local constraints. Berger [Berger, 1990] allows the snake to grow along features, and also to break.

Unfortunately, the convergence rate of a snake, using all points, is rather slow. Hence, some researchers [Fua and Leclerc, 1988; Amini *et al.*, 1988] use a polygonal approximation of the curve, but then smoothness can no longer be guaranteed. Another problem is that the only

way to include corners is to set $\alpha = 0$ at some locations, so the "cornerness" of the curve is not implicit.

A better way to simultaneously solve these problems is to use a parametric B-spline approximation of curves [Bartels *et al.*, 1987], as the next section shows. We call this new model a "B-snake".

3 B-SNAKES

In this model, the curve is replaced by its approximation by a B-spline and the energy of the approximation is minimized.

We first discuss the advantages of the scheme, then explain how to compute the B-spline approximation of the curve, and finally show the minimization procedure with B-snakes.

Let u be the parameter describing the approximating curve (we take u instead of s to remain consistent with notations in [Bartels *et al.*, 1987]), and $Q(u) = (x(u), y(u))$.

In this approximation, the curve is split into segments, and the joints between adjacent curve segments are called *knots*. Each curve segment is approximated by a piecewise polynomial function (order k), which is obtained by a linear combination of basis functions B_i and a set of control vertices $V_i = (X_i, Y_i)$:

$$Q(u) = \sum_{i=0}^{i=m} V_i B_i(u) \quad (5)$$

The control polygon can be calculated by performing a least-square fit of the data by the B-spline curve (paragraph 3.2).

In the following, let $p + 1$ be the number of points of the curve and $m + 1$ the number of vertices of the control polygon.

As is shown in paragraph (3.3), substituting v by $Q(u)$ in the snake energy equation (1) yields a similar system to (4), whose unknowns are the control vertices and therefore whose size is only $m + 1$ instead of $p + 1$.

3.1 Advantages of approximating B-splines

Local control : elementary B-splines B_i have local support, so that modifying the position of a data-point causes only a small part of the curve to change.

Continuity control : B-splines are defined with continuity properties at each point: order k B-splines are C^{k-2} continuous. But it is possible to control the continuity at the knots, by accepting multiple knots. These are obtained by letting successive knots be equal, which causes intermediate intervals to be empty. Let μ be the multiplicity degree of a knot, the continuity at this knot is then: $C^{k-1-\mu}$. When μ is equal to $k - 1$ the knot is C^0 continuous and the corresponding control point is interpolated by the curve.

This property is very interesting for the B-snake model: if we introduce a multiple knot whose degree of multiplicity is equal to $k - 1$, the first and second derivatives are no longer continuous at this

knot and the smoothness constraint is broken: a corner appears at this knot.

3.2 Control polygon

To find the control polygon at time 0, we perform a least-squares fit of the data by a B-spline curve [Bartels *et al.*, 1987; Saint-Marc and Medioni, 1990].

We want to minimize the distance between the discrete data points of the original curve and its approximation by a B-spline. This distance is given by the expression:

$$R = \sum_{j=0}^p |Q(u_j) - P_j|^2 = \sum_{j=0}^p ((x(u_j) - x_j)^2 + (y(u_j) - y_j)^2)$$

where u_j is some parameter value associated with the j^{th} data point, and $Q(u_j)$ is given by:

$$Q(u) = \sum_{i=0}^{i=m} V_i B_i(u) = \sum_{i=0}^{i=m} (X_i B_i(u), Y_i B_i(u)) \quad (6)$$

Since this equation is quadratic, the minima occurs for those values of X_i and Y_i such that:

$$\begin{cases} \frac{\partial R}{\partial X_l} = 0 \\ \frac{\partial R}{\partial Y_l} = 0 \end{cases}$$

where l ranges between 0 and m . So, we obtain:

$$\begin{aligned} \sum_{i=0}^m X_i \sum_{j=0}^p B_i(u_j) B_l(u_j) &= \sum_{j=0}^p x_j B_l(u_j) \\ \sum_{i=0}^m Y_i \sum_{j=0}^p B_i(u_j) B_l(u_j) &= \sum_{j=0}^p y_j B_l(u_j) \end{aligned} \quad (7)$$

This equation can be solved by a LU decomposition.

The choice of the number of vertices, $m+1$, determines how close to the original data the approximation is, which is measured by R . An automatic choice can then be performed [Saint-Marc and Medioni, 1990]: we set a fitting tolerance r_0 and we find the value of $m+1$ which yields the normalized distance $r = R/(p+1)$ closer to r_0 , using a binary search approach.

3.3 Minimization resolution

We want to minimize equation (1) by substituting the curve v by the analytical expression of its B-spline approximation (6).

The total energy of the curve is then:

$$\begin{aligned} E = \sum_{j=0}^p \{ & \frac{1}{2} \alpha(u_j) [(\sum_{i=0}^m X_i B_i'(u_j))^2 + (\sum_{i=0}^m Y_i B_i'(u_j))^2] \\ & + \frac{1}{2} \beta(u_j) [(\sum_{i=0}^m X_i B_i''(u_j))^2 + (\sum_{i=0}^m Y_i B_i''(u_j))^2] \\ & + F(v(u_j)) \} \end{aligned} \quad (8)$$

We are looking for control points coordinates X_i, Y_i that minimize E , that is, that satisfy:

$$\forall l \in \{0, \dots, m\} \begin{cases} \frac{\partial E}{\partial X_l} = 0 \\ \frac{\partial E}{\partial Y_l} = 0 \end{cases}$$

That yields for the X coordinate:

$$\begin{aligned} \sum_{j=0}^p [\alpha(u_j) B_l'(u_j) \sum_{i=0}^m X_i B_i'(u_j) + \\ \beta(u_j) B_l''(u_j) \sum_{i=0}^m X_i B_i''(u_j) + \\ B_l(u_j) \frac{\partial}{\partial x} F(\sum_{i=0}^m X_i B_i(u_j), \sum_{i=0}^m Y_i B_i(u_j))] \\ = 0 \end{aligned} \quad (9)$$

and a similar equation for Y . When we change the summation order, we get:

$$\begin{aligned} \sum_{i=0}^n X_i [\sum_{j=0}^p \alpha(u_j) B_l'(u_j) B_i'(u_j) + \\ \sum_{j=0}^p \beta(u_j) B_l''(u_j) B_i''(u_j)] \\ + \sum_{j=0}^p B_l(u_j) F_x(v(u_j)) = 0 \end{aligned}$$

for l ranges from 0 to m .

This equation set can be written in the same matrix form as (3), with $m+1$ equations of $m+1$ unknowns (X, Y) instead of $p+1$ (x, y):

$$\begin{cases} A_b X + G_x(x, y) = 0 \\ A_b Y + G_y(x, y) = 0 \end{cases} \quad (10)$$

where A_b is still a band matrix.

This system can be solved in way similar to the original snakes (4), and we have:

$$\begin{cases} X_{t+1} = (A_b + \gamma I)^{-1} (\gamma X_t - G_x(x_t, y_t)) \\ Y_{t+1} = (A_b + \gamma I)^{-1} (\gamma Y_t - G_y(x_t, y_t)) \end{cases} \quad (11)$$

4 APPLICATION : BUILDING TOPS DELINEATION FROM STEREO DATA

The detection of cultural features, such as roads and buildings in aerial imagery is an important application area in Computer Vision.

In recent work Fua [Fua and Hanson, 1989b; Fua and Leclerc, 1988] has proposed to detect such buildings by refining a coarse estimate through a parameter estimation phase. Mohan [Mohan and Nevatia, 1988] defines a building as a collation of rectangles and proposes to solve the selection process by a Constraint Satisfaction Network.

These methods use monocular information only, such as edges, to generate and verify hypotheses. When stereo data is available, they use it mostly in the verification stage to refine the estimates.

Here, we propose instead to use stereo first to guide in the detection of elevated structures, on the basis that their disparity is bound to be different from the disparity of the background, and to refine the estimates using monocular information.

Most stereo algorithms (see [Barnard and Fischler, 1982; Dhond and Aggarwal, 1989] for surveys) produce reliable results in images of rolling terrain, but degrade ungracefully when depth discontinuities occur, since the smoothness assumption becomes violated. This is true for area-based and feature-based methods. We use here an algorithm which combines both approaches, as described in [Cochran and Medioni, 1989]. The buildings roofs appear as regions of constant disparity, but their boundaries are very approximate, generally ragged.

We can refine them by using:

- monocular information: buildings are likely to generate intensity edges;
- smoothness: building boundaries are mostly smooth, with the exception of some corners;
- invariance: the boundaries should correspond in both images.

To turn these observations into a computational framework, we use the B-snakes described above. The internal energy captures the smoothness constraint, and we define an appropriate external energy for the other two constraints, as shown below.

To solve the problem introduced by corners, we proceed in two stages: first, the boundary is supposed smooth, and the snake reaches its convergence state, then potential corners are detected as extrema of curvature and the B-snake model is applied again.

We now give the details of the process and present some illustrative results.

4.1 Stereo energy

Kass [Kass *et al.*, 1988] applies snakes to the problem of stereo matching. According to some psychological evidence [Burt and Julesz, 1980], he assumes that, if two contours correspond then the disparity varies slowly along the 3-D contour. This constraint can be expressed in an additional energy functional:

$$E_{\text{stereo}} = (v^L(s) - v^R(s))^2$$

where v^L and v^R are left and right snake contours.

Fua [Fua and Hanson, 1989b] uses a stereographic effectiveness term which encodes the projected patch in the second image, while knowing its photometry in the first.

In our approach, the contours of non-null disparity areas are the first estimate of objects contours we want to improve, that is, the initialization of the snakes at time 0.

Furthermore, we can combine the left and right external energy of each object, by projecting the right one on the left one through the disparity map (equation 12). This allows us to filter non matching areas and to reinforce constraints in matched areas.

$$E_{\text{stereo}}(s) = E_L(s) + d(s)E_R(s) \quad (12)$$

Since edges are likely to correspond to depth or surface orientation discontinuities, we use edge information as monocular external energy. This energy supplies the feature-based information often used in stereo matching algorithm but which yields a sparse disparity map.

In order to increase the efficiency when the snake is too far from the edges, a distance map such as Chamfer distance [Barrow *et al.*, 1977] is added to the edge information.

4.2 Discontinuities C^0

Polygonal objects can be processed without a priori knowledge on their shape, by using a method in two steps:

1. First stage: Regular B-snakes are implemented and their energy is minimized until equilibrium.
2. Second stage: Corners are detected at points of maxima curvature (equation 13) and new B-snakes are implemented with multiple-knots at the corners. Then B-snakes converge from their previous state toward a new equilibrium.

$$\rho = \frac{x_u y_{uu} - x_{uu} y_u}{(x_u^2 + y_u^2)^{\frac{3}{2}}} \quad (13)$$

Images 2 and 3 show this process.

When corners are detected, we assume in this application, that polygonal objects are encountered. Then, new parameters are used in the second stage, to emphasize the behavior of the B-snake acting as a strong rod between corners.

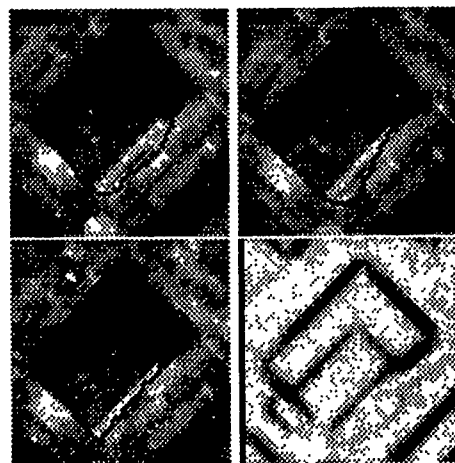


Figure 2: First example: Initialization, result of first step and final result. The external energy is also shown.

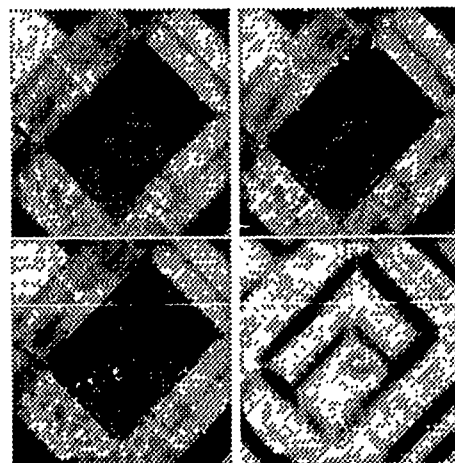


Figure 3: Second example: Initialization, result of first step and final result. The external energy is also shown.

4.3 Results

Images 4 to 17 show two series of examples obtained with quadratic B-snakes.

For each example, the left and right disparity maps are shown, which are blurred and noisy at corners. The initialization of B-snakes are extracted from the left one, and the process is performed on this side.

Results are shown at both steps of the process described above. We can see that after the first step, roofs borders are improved, but remained rounded at corners. They become sharper after the second step.

While it gives the direction of the nearest edge, the Chamfer distance helps the convergence especially when the curve is too far from the edges. But it does not provide reliable information at locations of multiple nearby edges. This and the lack of edge information at some locations contribute to cause B-snakes to stabilize into local minima.

Furthermore, this energy makes the B-snakes to shrink or to expand only if the first estimate is around local maxima otherwise, it shrinks until vanishing (for example: the highest tower cannot be handled considering the poor edge information used).

5 CONCLUSION

Snakes provide a tool to solve many vision problems by means of global energy-minimizing, while taking into account geometrical model of curves and image features information. As the energy is integrated along the entire length of the curve, it is less sensitive to image noise and various photometric anomalies.

We have improved this tool by using parametric B-spline approximations of curves that yield increasing convergence speed and allow the so-called *B-snake* to include corners.

Then, the B-snake can be applied to adjustment of non-smooth shapes. For example, it is able to refine the delineation of building tops from stereo aerial images, with a good accuracy, without using a priori knowledge or generic model.

Acknowledgements

We would like to thank J.L. Jezouin from Matra-SEP for providing the original stereo pairs used in this study.

References

- [Amini et al., 1988] A. Amini, S. Tehrani, and T. Weymouth. Using dynamic programming for minimizing the energy of active contours in the presence of hard constraints. In *Proceedings of 2nd International Conference on Computer Vision*, pages 95-99, Tampa, Florida, 1988.
- [Barnard and Fischler, 1982] S. Barnard and M. Fischler. Computational stereo. *ACM Computing Surveys*, 14(4):553-572, December 1982.
- [Barrow et al., 1977] H. Barrow, J. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 659-663, Cambridge, Massachusetts, August 1977.
- [Bartels et al., 1987] R. Bartels, J. Beatty, and B. Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*. Morgan Kaufmann, Los Altos, CA 94022, 1987.
- [Berger, 1990] M. O. Berger. Snake growing. In *First European Conference on Computer Vision*, pages 570-572, Antibes, France, April 1990.
- [Burt and Julesz, 1980] P. Burt and B. Julesz. A disparity gradient limit for binocular fusion. *Science*, 208:615-617, 1980.
- [Cochran and Medioni, 1989] S. D. Cochran and G. Medioni. Accurate surface description from binocular stereo. In *Proceedings of Workshop on Interpretation of 3D Scenes*, pages 16-23, Austin, Texas, Nov. 1989.
- [Dhond and Aggarwal, 1989] U. R. Dhond and J. K. Aggarwal. Structure from stereo-A review. *IEEE Transactions on Systems, Man & Cybernetics*, 19(6):1489-1510, November/December 1989.
- [Ferrie et al., 1989] F. Ferrie, J. Lagarde, and P. Whaite. Darboux frames, snakes, and super-quadratics: Geometry from the bottom-up. In *Proceedings of Workshop on Interpretation of 3D Scenes*, pages 170-176, Austin, Texas, Nov. 1989.
- [Fua and Hanson, 1989a] P. Fua and A.J. Hanson. Objective function for feature discrimination theory. In *Proceedings of the DARPA Image Understanding Workshop*, pages 443-460, May 1989.
- [Fua and Hanson, 1989b] P. Fua and A.J. Hanson. An optimisation framework of feature extraction: Applications to semiautomated and automated feature extraction. In *Proceedings of the DARPA Image Understanding Workshop*, pages 676-694, May 1989.
- [Fua and Leclerc, 1988] P. Fua and Y. G. Leclerc. Model driven edge detection. In *Proceedings of the DARPA Image Understanding Workshop*, volume 2, pages 1016-1021, Cambridge, Massachusetts, April 1988.
- [Kass et al., 1988] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, 1:321-331, January 1988.
- [Mohan and Nevatia, 1988] R. Mohan and R. Nevatia. Perceptual grouping for the detection and description of structures in aerial images. In *Proceedings of the DARPA Image Understanding Workshop*, pages 512-526, Boston, Massachusetts, April 1988.
- [Saint-Marc and Medioni, 1990] P. Saint-Marc and G. Medioni. B-spline contour representation and symmetry detection. In *First European Conference on Computer Vision*, pages 604-606, Antibes, France, April 1990.
- [Zucker et al., 1988] S. Zucker, C. David, A. Dobbins, and L. Iverson. The organization of curve detection: Coarse tangent fields and fine spline coverings. In *Proceedings of 2nd International Conference on Computer Vision*, pages 568-577, Tampa, Florida, Dec. 1988.

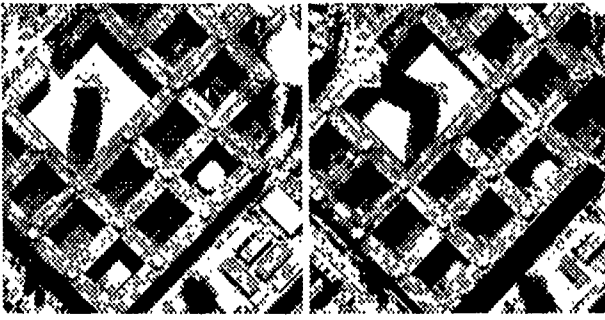


Figure 4: First example: stereo intensity images

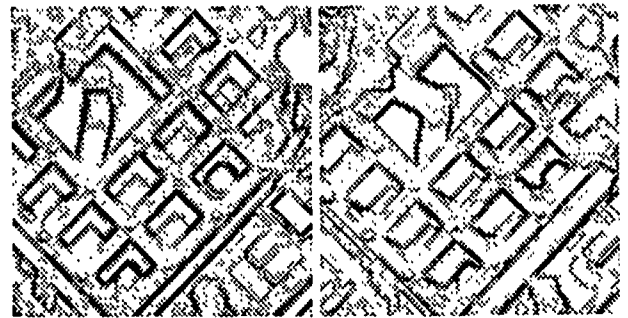


Figure 8: Left and right final energies

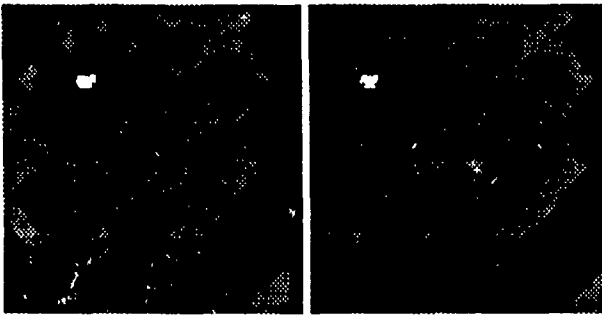


Figure 5: Left and right disparity map

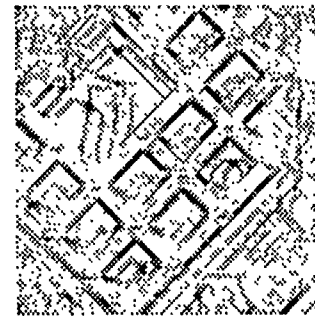


Figure 9: Global stereo energy



Figure 6: Left and right negated gradient

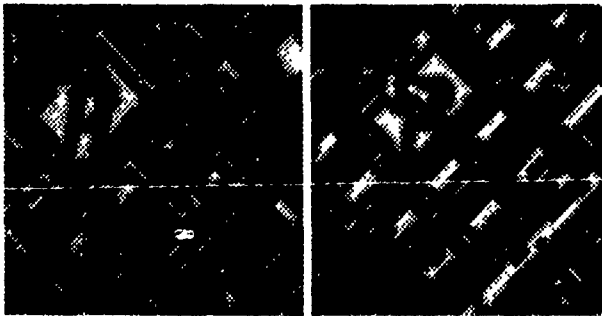


Figure 7: Left and right Chamfer distance

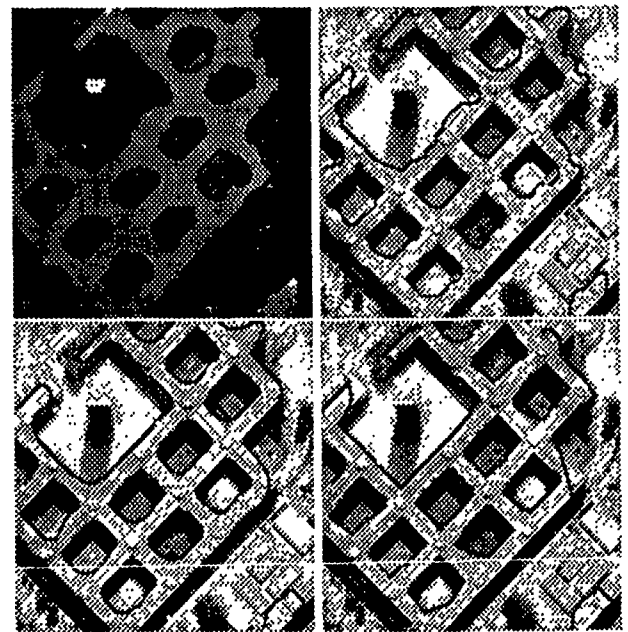


Figure 10: Different steps of delineation of buildings roofs from the first estimate of B-snakes from edges of disparity map to the final result with corners

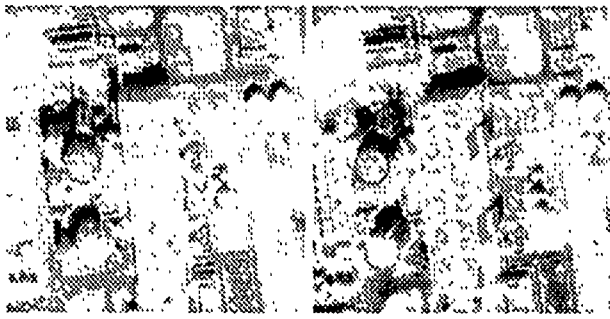


Figure 11: Second example: stereo intensity images

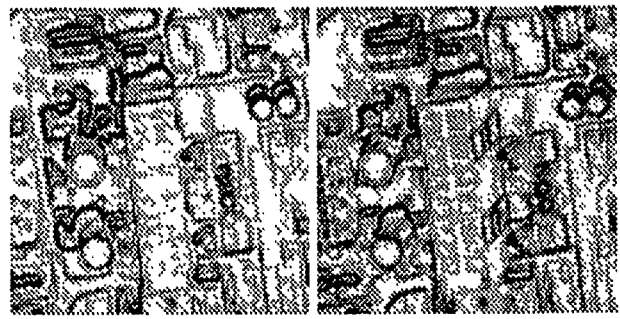


Figure 15: Left and right global energy

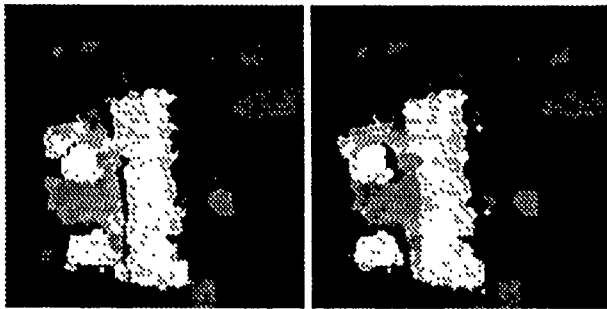


Figure 12: Left and right disparity

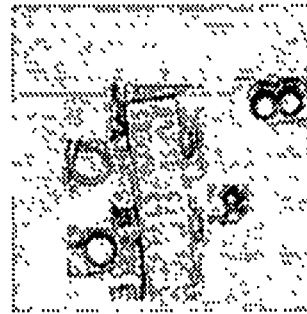


Figure 16: Global stereo energy



Figure 13: Left and right negated gradient

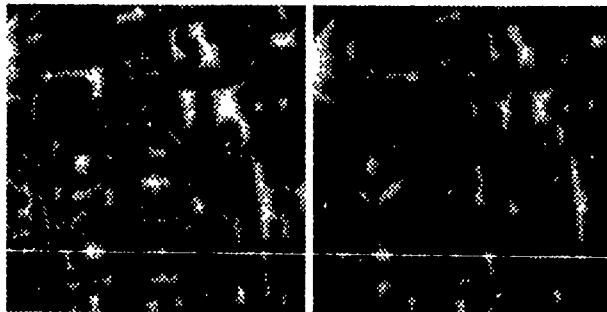


Figure 14: Left and right Chamfer distance

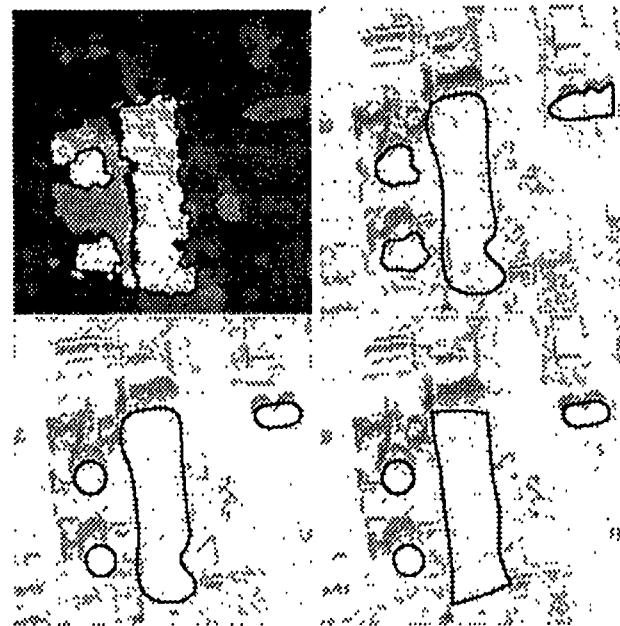


Figure 17: Different steps of delineation of buildings roofs from the first estimate of B-snakes from edges of disparity map to the final result with corners

BENCHMARK EVALUATION OF A MODEL-BASED OBJECT RECOGNITION SYSTEM

A. J. Heller and J. L. Mundy *

GE Corporate Research and Development Center
Schenectady, NY 12345

Abstract

A benchmark evaluation test of a model-based recognition system is discussed. The system was tested on a series of aerial reconnaissance images to evaluate recognition performance on the task of airfield monitoring. The effectiveness of the model pose constraint for recognition is discussed as well as an approach for selection of model features. The use of distance transforms for model hypothesis confirmation is also discussed.

1 Introduction

It is rare that an object recognition system reaches any degree of maturity. In most cases, a system is implemented to demonstrate a particular concept and the effort terminates after a few experiments and publication of the results. The recognition system discussed in this paper has been under continuous development for over five years and we have recently carried out a benchmark test of the system in the context of aerial reconnaissance. The following paper describes the design of the system and documents our experience with the model-based object recognition approach.

The initial version of the vertex-pair matching system was completed in 1986 and was able to recognize three-dimensional polyhedral objects in reasonably cluttered scenes with partial occlusion [Thompson and Mundy, 1987b]. Since that time, the system has been under continuous development and is currently being evaluated on the reconnaissance task of airfield monitoring. We are conducting a benchmark test of the system which presents a database of images taken under a wide variety of weather and viewing conditions. Major improvements have been made since the original implementation of the system to provide automatic model feature selection and match hypothesis confirmation. In the following sections we describe the current design of the vertex pair matcher and summarize the initial results of the benchmark test. The benchmark results provide statistical

*Work at GE was supported in part by the DARPA Strategic Computing Vision Program in conjunction with the Army Engineer Topographic Laboratories under Contract No. DACA76-86-C-0007 and the Air Force Office of Scientific Research under Contract No. F49620-89-C-0033.

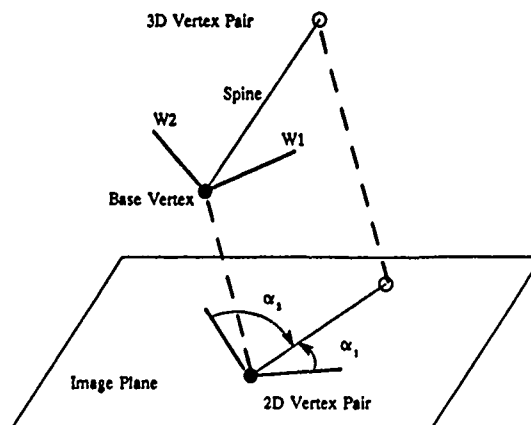


Figure 1: The vertex-pair geometry.

data which will be useful in evaluating theories about the effectiveness of filtering on transformation parameters [Grimson and Huttenlocher, 1990].

2 The Vertex-Pair Matcher

2.1 The Vertex-Pair Geometry

The vertex-pair configuration is shown in Figure 1 which illustrates the projection of the three-dimensional structure onto the image plane. The three-dimensional structure and its two-dimensional projection is referred to as a vertex-pair. The structure consists of two vertices, each defined by the intersection of two edges in the object model. Associated with one vertex are the two edges that define the one of the vertices. These edges provide orientational constraints which are exploited in computing the model-to-image transformation.

The edge orientation is defined in terms of the unit direction vectors of the object coordinate system. Note that the line joining the two vertices, called the *spine*,

does not have to correspond to an edge in the object model. It serves only as an reference for orientation and scale. The angles between the edges and the spine are projected into the image plane as α_1 and α_2 , as shown in Figure 1.

In effect, the vertex-pair provides a coordinate reference frame for the object which is easily related to the results of standard image segmentation algorithms. Image vertices are usually defined by the intersection or junction of image line segments. Thus, any vertex will have at least two incident edges. All vertices which are defined by the intersection of edges which adequate orientational accuracy are grouped pairwise to form the vertex-pair image feature. Note that vertex position and edge orientation does not depend on having complete projections of the object edges, making the image vertex-pair geometry somewhat immune to occlusion.

In the absence of any other constraints, the grouping of vertices into vertex-pairs is carried out for all pairs in the image. The grouping of n vertices results in $2kn^2$ vertex-pairs, since the orientation can be determined from either of the two vertices of the vertex-pair. The scalar, k , measures the number of combinations of edge pairs for vertices that have more than two edges. The factor of 2 arises since either edge of the vertex can be assigned to the corresponding model edge. The number of pairs can be large; in our recent experiments with high-resolution images, we find that the number of vertex-pairs is on the order of the number of pixels in the image. However, the subsequent clustering operations on each vertex-pair are simple and can all be carried out in parallel [Thompson and Mundy, 1987a].

2.2 Determining Transform Parameters

The next issue is the determination of the transformation between the model and the image reference frames, given an assignment of a model vertex-pair to an image vertex-pair. Determining this transformation is greatly simplified by making an affine approximation to the perspective mapping [Thompson and Mundy, 1987b]. An orthographic projection with a scale factor is an *affine* transformation. This same approximation was observed by Roberts[1965], which he calls "weak" perspective. The affine transformation applies to viewing situations where the maximum diameter subtended by the object is small, compared to the distance from the object to the camera.

The transformation between the object and the image coordinate reference frames has six degrees of freedom, three for translation and three rotations. The vertex-pair provides a sufficient number of constraints to fully determine the six parameters of the affine transformation.

Assume that a correspondence has been made between the affine projection of a 3D model vertex-pair and a set of 2D edges and vertices derived from the image intensity data. The tip and tilt rotations of the model about the x-y axes of the viewplane can be determined directly from the angles between the projected edges and spine of the vertex-pair, α_1 and α_2 . In our implementation, the mapping between α_1 and α_2 and tip and tilt orientation, ϕ and ψ , is precomputed for each model vertex-pair.

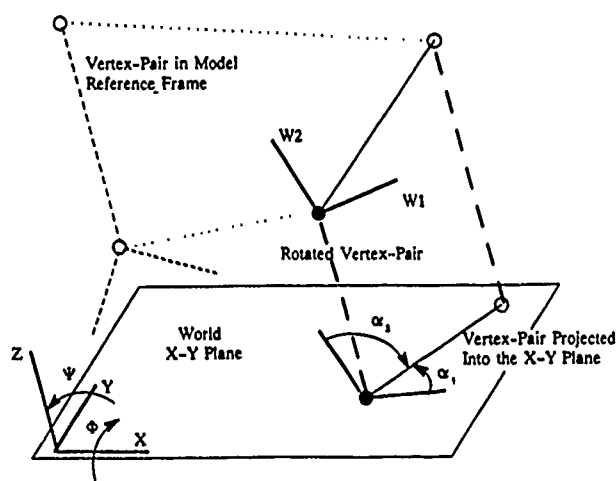


Figure 2: The projected angles α_1 and α_2 determine the camera tip and tilt rotations of the model vertex pair projected onto the image plane.

Once the tip and tilt angles are known, the model vertex-pair can be orthographically projected into the image plane as shown in Figure 2. The rotation about the axis perpendicular to the viewplane is simply the angle between the projected model spine and the spine of the image vertex-pair as illustrated in Figure 3. The image plane components of translation are just the 2D translation between the projected base vertex of the model vertex-pair and the corresponding image vertex pair. The third translational component, depth, is represented as an affine scale factor (ratio) computed from the projected model spine length and the image spine length. The spline length ratio can be converted to depth units if the camera focal length is known. Figure 4 shows the relationship between scale factor and depth. The procedure, just outlined, results in the complete evaluation of the transformation between the model and the image projection, given a single assignment between a vertex-pair in the image and a corresponding vertex-pair in the object model.¹ The next issue is to establish a method for determining the validity of such assignments.

2.3 Transform Parameter Clustering

2.3.1 Validity Through Clustering

The previous section has outlined a procedure for determining the six element transformation between the model vertex-pair and a corresponding pair of image ver-

¹We note that an equivalent set of geometric constraints is produced if a single edge is associated with each vertex instead of grouping both edges at one vertex. This configuration of two vertices, each with an orientation, is a well known affine coordinate basis.

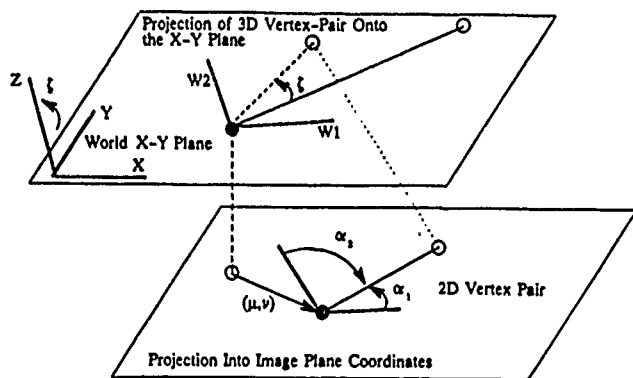


Figure 3: Image translation and rotation about the camera axis are determined by vertex position and spine orientation.

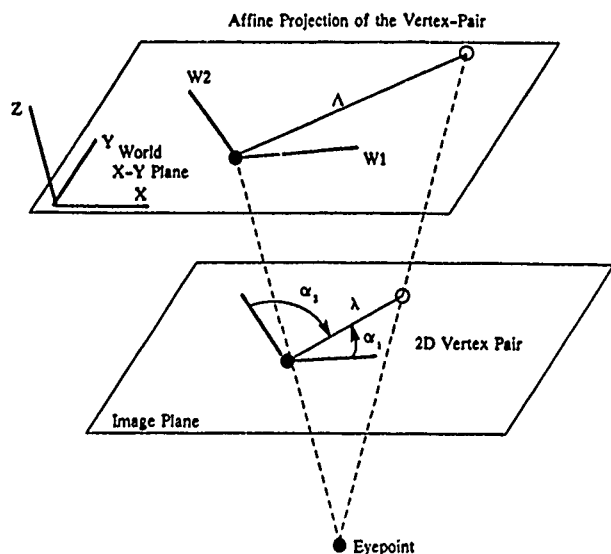


Figure 4: The relationship between spine lengths and viewing distance.

tices. This transformation produces a mapping of a single vertex-pair onto the image coordinate system. However, there is no guarantee that this assignment and the resulting transformation actually corresponds to a valid projection of the complete model into the image.

The basis for validity is to compare the transformation computed from a given assignment with those of other assignments. If the object is assumed to be a rigid body, then all valid assignments should result in transformations that are close in the six dimensional space, according to the principle of view consistency.

In a number of other systems, a match of a key model feature, such as a long edge, or a set of parallel lines, is used to establish an initial transformation. Subsequent assignments are added which are consistent with the initial transformation and the transformation parameters of the evolving group are updated accordingly [Grimson and Lozano-Perez, 1985, Ayache and Faugeras, 1984, Lowe, 1985].

One of our main assumptions is that, due to segmentation errors, there are no special or key features available to provide an initial transformation to filter the validity of other assignments. Instead, the validity of a transform is determined solely on the number of assignments that produce similar transform values [Ballard, 1981, Silberberg *et al.*, 1986].

This grouping of transforms is known as "binning" or "voting" in transform space. The use of voting as a basis for the validity of assignments follows from the conclusion that the fragility of existing image segmentation techniques prevents single key features from being reliably grouped. It is quite likely that features are missed due to occlusion, shadows, or low contrast; even the simple vertex-pair group cannot always be reliably detected in images with unconstrained lighting and viewpoint. It is assumed that valid assignments produce transform values that are "near" each other in transformation space. The definition of "near" depends on the amount of error expected in computing the transform values from the image. Our experiments indicate that valid clusters can have an extent of at least 10 degrees in angle and 10 pixels in translation.

2.3.2 Decomposing Transform Space

For convenience, we factor the six parameter transform space into four sub-spaces $[\phi, \psi]$, $[\zeta]$, $[\mu, \nu]$, and $[w^{-1}]$ (i.e. [tip, tilt], [rotation about the camera axis], [translation in the image plane] and [image scale]). The initial $[\phi, \psi]$ clustering is accomplished by sorting into a two-dimensional bin array set at 2 degree intervals. The sequence of clustering steps is shown in Figure 5. The basis for this choice of groups is primarily for ease of computation. The transform points are first assigned into $[\phi, \psi]$ bins to begin the search for clusters.

In order to account for errors in the computation of $[\phi, \psi]$ from the image features, it is desirable to include a range of solutions from the $[\alpha_1, \alpha_2] \rightarrow [\phi, \psi]$ map. A 3×3 region around the measured $[\alpha_1, \alpha_2]$ pair is extracted from the table. This gives a range of 15 degrees about the measured values since the table is computed in increments of 5 degrees. The resulting collection of $[\phi, \psi]$

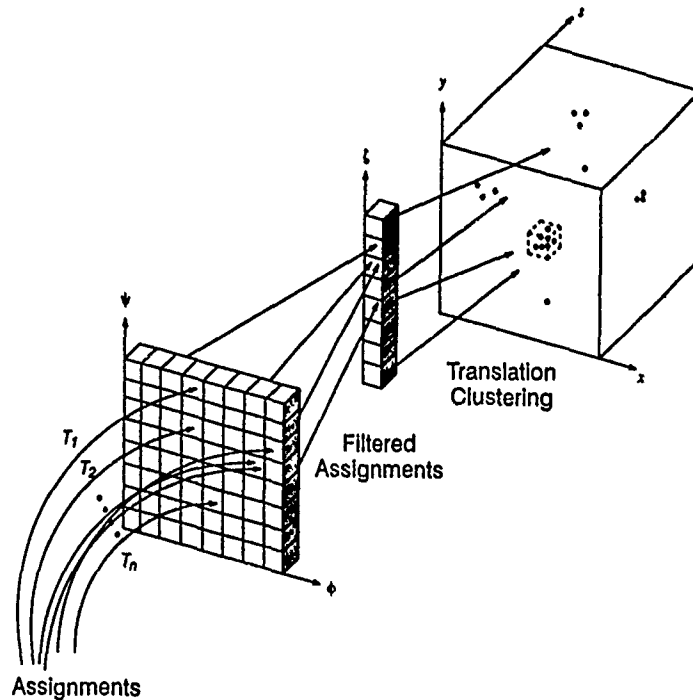


Figure 5: The clustering of transform parameters is carried out in stages using a combination of binning and nearest neighbor clustering techniques.

values is assumed to comprise the feasible rotations that could have produced the measured vertex-pair angles. It is important to allow this range of multiple inverse solutions since small variations in projected image angles can lead to large variations in $[\phi, \psi]$. Some viewpoints produce a degenerate relationship between the parameters of the transformation and the model to image correspondence. In this case, large rotational or translational errors do not produce correspondingly large errors in the location of projected model features. This projection sensitivity is an important criterion for selecting model vertex-pairs and will be discussed in more detail later.

The bin array is scanned for clusters by sweeping a 3×3 array over the histogram. If the number of samples in the 3×3 array exceeds a minimum number (say 4) of assignments then that cluster is used to form a one dimensional histogram with respect to ζ . This one dimensional array of bins is scanned with a 1×3 window and if a sufficient number of samples are found in the window, then they are declared to be a cluster. The resolution of the ζ bins is also 2° .

The grouping in $[\mu, \nu, w^{-1}]$ space is carried out using a nearest neighbor clustering algorithm. Any point in a valid cluster formed in rotation space is used to start a cluster in $[\mu, \nu, w^{-1}]$ space. The euclidean distance from the initial point to subsequent assignments in the rotation cluster is calculated. If the distance is smaller than some allowable tolerance then the point is added to the growing cluster. If a point lies outside the tolerance radius, it is rejected and used to start a new cluster. A

typical tolerance radius used in the current experiments is 10 pixels in translation and a factor of 20% in scale.

The final group of transform clusters is further refined by eliminating improper assignments. An image vertex-pair can be improperly assigned to two or more model vertex-pairs. It is also improper to allow assignments that are not mutually visible from the viewpoint corresponding to the transform of the cluster. This condition is prevented by considering visibility when computing the $[\phi, \psi]$ map. At this point, a cluster with four or more assignments is considered to be a feasible hypothesis for an instance of a projection of the model in the image.

2.4 Search Space Reduction

In the development of the vertex-pair matcher no assumptions have been made about the viewpoint or the parameters of the camera with which the images are made. In reality, in almost all reconnaissance applications the location and orientation of the camera and its parameters (e.g. image plane size and focal length) are known with a great deal of precision. In addition the poses of the objects to be recognized are subject to well known constraints (i.e. an airplane is parked resting on its wheels as opposed to being balanced on its nose; buildings have their bases coincident with surrounding terrain.) All of this information can be used to reduce the amount of computation and increase the robustness of the algorithm.

In practice we derive the location of the sensor plat-

form by identifying known ground control points in the image and running the USGS Space Resection algorithm [Slama, 1980] on it. This algorithm computes the camera location and orientation, given the coordinates of the corresponding pairs of ground control points and image points and the camera parameters. This information is then used to define a unit up-vector perpendicular to the ground plane at the center of projection of the image.

Due to perspective distortion a correction vector must be added to the up-vector at other points in the image. This vector is calculated by finding the vector from the center of projection of the image to the image point under consideration and dividing its magnitude by the focal length of the lens. (If the image has been made with a long focal length lens, the uncorrected up-vector is a good approximation for the up-vector throughout the image plane.) We also define a unit up-vector for each model. The degree of alignment is then gauged by taking the dot-product of the two vectors. Typically we would consider a value greater than 0.9 to indicate an plausible pose. This translates to an angular deviation of less than 25 degrees.

This constraint is then used in two places in the matcher. The first is while computing the $[\alpha_1, \alpha_2] \rightarrow [\phi, \psi]$ maps. Only values of ϕ and ψ that could produce acceptable alignments at any point in the image and for any ζ are entered into the map. This eliminates the calculation of transforms for assignments that could not produce acceptable model poses.

The second is after transform calculation. Only those assignments which produce acceptable transforms are passed on to the clustering stage of the matcher. In addition, at this point, assignments are also filtered on scale. Only those having scale values consistent with the information about the camera model are used for producing clusters. Again, due to perspective distortion, the scale of a given object can vary due to its location on the image plane and this range of variation must accounted for in the acceptance function.

3 Model Feature Selection

Initial experiments with transform clustering indicated the need to account for the errors associated with the placement of features in the image segmentation. We now consider these issues in more detail.

3.1 Weaknesses of this Approach

It is clear that the success of the clustering process is crucially dependent on the accuracy with which the transformation can be determined from each vertex-pair assignment because we depend on having a small cluster radius threshold in transform space to filter out incorrect matches.

Reliable cluster detection requires good accuracy in the location and orientation of image features since for some viewpoints, the uncertainty in these feature parameters is amplified in the calculation of transformation parameters.

The current implementation accounts for this uncertainty in the transform values by including all transform

solutions which would correspond to an assumed spread in the image feature parameters. For example, we assume that the projected angles, α_1 and α_2 , can be in error by 5° due to uncertainty in the image segmentation process. This propagated uncertainty in the model transformation can lead to a range of solutions for the model match. Each solution corresponds to a slightly different value for the rotation parameters of the model transformation.

The relationship between image segmentation errors and the model transformation uncertainty depends on the viewpoint. It is also the case that degeneracies exist in the viewing projection. In particular transformation the equations become degenerate for viewpoints which are collinear with the edges and spine of the vertex-pair.

In the original implementation, the model vertex-pairs were manually selected with an interactive graphics editor which operates on the object model. There is no guarantee that the vertex-pairs selected in this manner will exhibit good error performance. Since it is essential that the set of vertex-pairs selected provide as accurate an estimate as possible of the model transform parameters over all viewpoints, we have extended the implementation to satisfy the following goals:

- Establish an error measure for the model transformation which can serve as a cost function in optimizing the selection of model vertex-pair features.
- Clarify the nature of viewing degeneracies for the vertex-pair feature, so that these viewpoints can be avoided in the clustering process.
- Implement an optimization algorithm to automatically select vertex-pairs.

The remainder of this section is devoted to a discussion of these issues and includes some experimental results obtained for an error measure which appears to satisfy the criteria just stated.

In the discussion to follow we focus on the determination of (ϕ, ψ) . The error sensitivities of the other parameters of the transformation depend mainly on foreshortening and scale which are also functions of (ϕ, ψ) . The effects of viewing distance and viewing orientation are thus mixed, which leads to the idea of multi-resolution model feature sets.

The key issue is how the vertex-pair is projected as the viewpoint moves over the surface of the standard *view-sphere*, which is a spherical surface defined to represent the possible viewing orientations of the image plane with respect to the object. The object is considered to be at the origin of the sphere and the points on the surface of the sphere define a vector orientation from a given point to the origin. Any location on the viewsphere can be directly represented in terms of the camera tip-tilt orientations (ϕ, ψ) .

For each point on the viewing sphere, there should be an adequate number of potentially visible vertex-pairs which have acceptable error performance and are not degenerately viewed for that viewing orientation. Naturally, one can not guarantee that a given vertex-pair will be visible, since it may be occluded by other objects, or by part of the object surface itself. In our current im-

plementation, the local self occlusion of a vertex-pair is taken into account by not allowing occluded viewpoints to appear as solutions in the $[\alpha_1, \alpha_2] \rightarrow [\phi, \psi]$ map. We do not solve the full hidden line problem to determine global occlusion, although this is desirable for a full solution to the optimization problem.

Next we consider the definition of an appropriate error measure which describes the uncertainty in the (ϕ, ψ) parameters as a function of (ϕ, ψ) orientation, or equivalently, with respect to position on the viewsphere.

3.2 Image Segmentation Error

In our current approach to segmentation [Canny, 1983], we rely on zero crossings of the second derivative of image intensity to define the location of geometric edge and vertex features. There are many phenomena that can cause the location defined by the second derivative to be in error with respect to the ideal location corresponding to the image projection of a given object feature. Some of the more significant effects:

- Complex image intensity behavior that does not correspond to the simple step edge model employed to detect object boundaries (e.g. corners and junctions).
- Boundary characterized by texture, rather than simple intensity discontinuity.
- Quantization in image intensity and spatial resolution.
- Random uncertainty in sensor intensity values (Usually a small effect).

These effects contribute to uncertainty in the detected boundary element locations. In our approach these "edges" are then linked, and the resulting boundary chain is approximated by straight line segments [Asada and Brady, 1984]. This process introduces additional error in the segmentation geometry. Some of the major effects here are:

- Boundary chains with low curvature do not have well defined segment endpoint location.
- Image spatial quantization introduces significant uncertainty in the chain curvature measurement.

Finally, even more uncertainty is introduced by the necessary extrapolation of line segments to form vertices between endpoints of lines that should (ideally) meet. This extension is necessary because some portions of the boundary are missing due to poor edge detector performance near junctions and in the case of low contrast boundary intervals.

The cumulative result of these various phenomena in our current implementation is that edge orientation is accurate to about 5° , and vertex location is accurate to a radius of several pixels. These errors can be much larger for the case of line segments with lengths that are comparable to the vertex uncertainty (5-10 pixels).

The error in edge orientation is the focus of our discussion in which we consider the relationship between this orientation error and the (ϕ, ψ) parameters.

3.3 A Rotation Error Measure

The equational systems that relate (α_1, α_2) and (ϕ, ψ) are non-linear and present many special solution cases. In our approach, these equations are solved numerically and stored in lookup tables. We define a Taylor series expansion about a particular value of (ϕ, ψ) to provide a linear expression for the parameter mapping.

We assume that we have computed the functions, $\alpha_1(\phi, \psi)$ and $\alpha_2(\phi, \psi)$ (see Figure 1). Then,

$$\alpha_1(\phi, \psi) = \alpha_1(\phi_0, \psi_0) + \frac{\partial \alpha_1}{\partial \phi}(\phi - \phi_0) + \frac{\partial \alpha_1}{\partial \psi}(\psi - \psi_0) \quad (1)$$

$$\alpha_2(\phi, \psi) = \alpha_2(\phi_0, \psi_0) + \frac{\partial \alpha_2}{\partial \phi}(\phi - \phi_0) + \frac{\partial \alpha_2}{\partial \psi}(\psi - \psi_0) \quad (2)$$

where the indicated derivatives can be computed numerically.

The Jacobian, J , of the parameter mapping is given by,

$$J = \begin{vmatrix} \frac{\partial \alpha_1}{\partial \phi} & \frac{\partial \alpha_1}{\partial \psi} \\ \frac{\partial \alpha_2}{\partial \phi} & \frac{\partial \alpha_2}{\partial \psi} \end{vmatrix} \quad (3)$$

Naturally, if J vanishes, then the mapping is not defined for that particular viewpoint [Whitney, 1955]. We can solve the expansion equations for the variation in (ϕ, ψ) as follows:

$$\Delta \phi = \frac{\Delta \alpha_1 \frac{\partial \alpha_2}{\partial \psi} - \Delta \alpha_2 \frac{\partial \alpha_1}{\partial \psi}}{J} \quad (4)$$

$$\Delta \psi = \frac{\Delta \alpha_2 \frac{\partial \alpha_1}{\partial \phi} - \Delta \alpha_1 \frac{\partial \alpha_2}{\partial \phi}}{J} \quad (5)$$

Assuming that the errors in (α_1, α_2) are independent and of zero mean, we can derive an estimate for the variance in the Euclidean distance between the (ϕ, ψ) estimate and the mean, (ϕ_0, ψ_0) . We denote this squared distance by $\sigma_{\phi, \psi}^2$. A similar representation can be defined for the variance in the measured projection angles, i.e. $\sigma_{\alpha_1, \alpha_2}^2$. The ratio of these variances is given by,

$$\frac{\sigma_{\phi, \psi}^2}{\sigma_{\alpha_1, \alpha_2}^2} = \frac{\frac{\partial \alpha_2}{\partial \psi}^2 + \frac{\partial \alpha_1}{\partial \psi}^2 + \frac{\partial \alpha_1}{\partial \phi}^2 + \frac{\partial \alpha_2}{\partial \phi}^2}{J^2} \quad (6)$$

3.4 Effective Viewing

We are now in a position to define the concept of effective viewing. An effective vertex-pair is one which provides a precise estimate of the coordinate transform between the three-dimensional reference frame and the image projection reference frame.

A single vertex-pair cannot be effective over all viewpoints. There are degenerate viewing conditions where the transformation,

$$(\alpha_1, \alpha_2) \rightarrow (\phi, \psi)$$

is not defined. For example, if the viewing direction is collinear with the spine or either of the two edges, the corresponding transform equations do not have a solution.

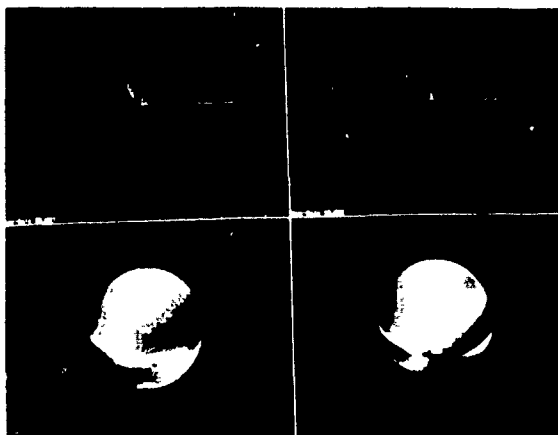


Figure 6: Two examples of the error measure projected onto the viewing sphere. The upper panes show the vertex-pairs used to generate the spheres shown in the lower pane.

The error measure that we have just defined predicts these degeneracies as illustrated in Figure 6. In this figure, we have projected $\frac{\sigma_{\phi\psi}^2}{\sigma_{\alpha_1\alpha_2}^2}$ onto the viewing sphere. In the left column, a coplanar vertex-pair is shown in the upper pane. The spine is horizontal and the displayed viewpoint is somewhat above the plane of the vertex-pair. One of the edges is quite foreshortened at this viewpoint. The viewsphere for this case is shown in the lower left pane. The eyepoint for this image of the sphere is the same as that of the vertex-pair in the upper pane. The intensity in this image is proportional to the ratio, $\frac{\sigma_{\phi\psi}^2}{\sigma_{\alpha_1\alpha_2}^2}$. The error measure becomes high at the equator of the viewsphere, since this corresponds to the locus of viewpoints that lie in the plane of the vertex-pair. The edges and spine of the vertex-pair all collapse into a single line for this set of viewpoints. Singularities exist at the points the view axis is collinear with any of the edges or the spine.

There is also a region of relatively high error at the poles of the viewsphere. These viewpoints correspond to looking normal to the plane of the vertex-pair. This is reasonable, since the projected angles of edges lying in a plane, have the least variation with tip and tilt of the plane when the viewpoint is normal to the plane.

The right column shows the same calculation except that the vertex-pair is not coplanar. The first edge is now inclined upward out of the horizontal plane at 45°. There are still two great circles of high value on the viewsphere. These correspond to viewpoints lying in the two planes defined by the spine and each of the edges. The error associated with normal views of each of these planes now contributes to a rather complex distribution over the sphere. The dark areas on the sphere correspond to effective viewpoints.

3.5 Scale and Translation Errors

A further refinement of the error measure includes translation and scale errors. Since scale is computed as the ratio of the spines in the scene and model vertex-pairs, the error propagates through the transformations in inverse proportion to the spine length of the 3d vertex-pair. This has been observed subjectively in manually selected vertex-pairs – those with longer spines yield more accurate transformations.

Propagation of translation errors is directly proportional to the distance of the base vertex from the origin of the model. This would suggest that all models should have their origin placed to minimize the distance to the vertices so that the selection of vertex-pairs is not biased by the relatively arbitrary placement of the model origin.

3.6 The Composition of Multiple Vertex-Pairs

If we assume that the variation in the projected angles of vertex-pairs is due to statistically independent segmentation error, then a composite error measure for a set of vertex-pairs can be defined. That is,

$$\left\langle \frac{\sigma_{\phi\psi}^2}{\sigma_{\alpha_1\alpha_2}^2} \right\rangle = \frac{1}{N} \sum_{i=1}^N \left(\frac{\sigma_{\phi\psi}^2}{\sigma_{\alpha_1\alpha_2}^2} \right)_i \sigma_{\alpha_1\alpha_2}^2 \quad (7)$$

Where $\sigma_{\alpha_1\alpha_2}^2$ is the expected variance in the angles associated with a particular vertex-pair in the segmentation. In general, the variance of edge angles in the segmentation is not directly related to the associated model vertex-pair. The accuracy of angle determination between image boundary segments is inversely proportional to edge length. The length is controlled by occlusion and edge contrast effects, and is not closely related to the projected edge length of the ideal object edge.

Therefore we can simplify the composition process by assuming that the variance $\sigma_{\alpha_1\alpha_2}^2$ is the same for all vertex-pairs grouped from the segmentation, and further that this variance is the worst case value corresponding to the shortest acceptable segmentation edges. The resulting composite error measure is simply the average of the error measure for the constituent vertex-pairs.

3.7 Automatic Vertex-Pair Selection

A major application for the vertex-pair error measure is in automatic vertex-pair selection. The goal of this process is to select a set of model vertex-pairs so from any viewpoint there is some minimum number (say 6) model vertex-pairs which are visible and possess good performance *from that viewpoint*. Given the large number of vertices in the model, it is necessary to avoid selecting a correspondingly large number of model vertex-pairs. Using an excessively large number of vertex-pairs is undesirable since complexity of the algorithm is linear in the number of vertex-pairs used. Therefore we want to use the smallest number of vertex pairs consistent with good error performance.

Candidate vertex-pairs are evaluated with three criteria. The first criteria arises from a simple visibility model. Those vertex-pairs which are occluded by other parts of the model over most of the viewsphere are not considered.

The second uses a simple sensor model. Short object edges are not likely to appear in the segmentation of an image and those that do cannot provide accurate position or orientation information. Because of this, vertex-pairs in the model with short spines or edges are not likely to participate in correct clusters and therefore are eliminated from consideration.

Finally, the error performance of each vertex-pair is examined. A vertex-pair with either edge nearly collinear with its spine is known to have high error over much of the viewsphere so these are eliminated. The remaining vertex-pairs are evaluated with the error measure over the area of viewsphere over which they are visible. We define the coverage area of a vertex-pair to be the surface area of the viewsphere which is visible and of low error.

To generate the set of model vertex-pairs, first the vertex-pairs with the greatest coverage are selected. The remaining vertex-pairs are again sorted by the area of the viewsphere each covers which is not covered by the required number of pairs previously chosen. This is continued until either the entire view sphere is adequately covered or none of the remaining vertex-pairs provide any new coverage. The later is the more common situation, however those remaining areas of the view sphere are still covered by vertex-pairs which exhibit high error there.

3.8 An Example

The automatic vertex-pair selection algorithm picked 31 vertex-pairs to characterize the model of the C-130 transport aircraft used in the benchmark. Figure 7 shows the model and the composite error spheres rendered from the same viewpoint. The upper left-hand sphere shows the error performance of the single vertex-pair shown superimposed on the model (it groups the forward wingtip vertex with the vertex where the tail meets the body and the edges along the leading edge of the tail and the top of the body). Brighter areas on the viewsphere correspond to areas of poor error performance. It shows that the error performance of an individual vertex-pair can be highly complex and not related in an obvious manner to the geometry of the vertex-pair. The lower right sphere shows the depth of coverage of vertex-pairs on the model. Lighter areas represent areas of greater coverage. The lower left sphere shows the areas of the viewsphere covered by at least six vertex-pairs. It can be seen that the automatic selection algorithm has been able to cover almost all of the viewsphere with low-error vertex-pairs.

4 Match Confirmation

The criterion of view consistency is not sufficient to reliably detect model instances in large and cluttered images. The probability of forming large transform clusters from random feature assignments grows rapidly with the number of features in the image, as observed by Grimson and Huttenlocher [1990]. We have verified their general predictions on the poor filtering effectiveness of rotational parameters alone. The overall effect of filtering on all six transformation parameters is still quite effective, however. In a typical case, removing all clusters

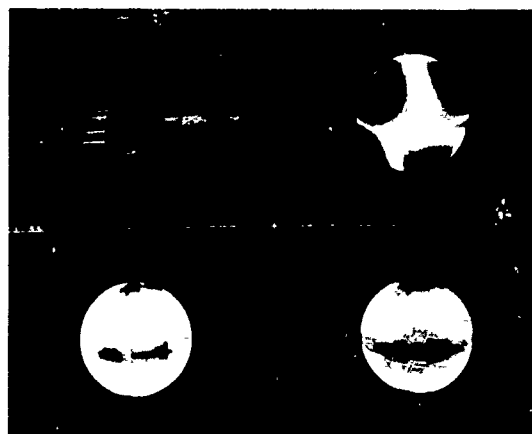


Figure 7: The C-130 model and its error spheres rendered from the same viewpoint.

with four or less assignments produces a thousand-fold reduction in the number of feasible model hypotheses without eliminating any actual model instances in the image. However the number of remaining clusters is still large, on the order of 10^4 , and must be significantly pruned by testing the image feature support given to the hypothesized position and orientation of model.

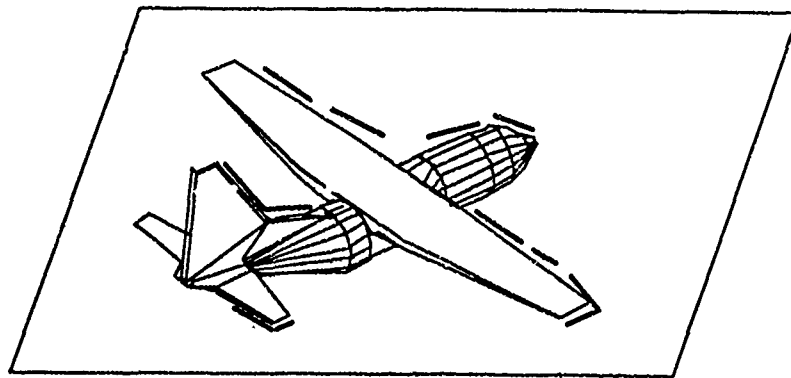
This process of match confirmation carried out by comparing the model edges which are predicted to appear in the image with all of the edge segments actually recovered by segmentation. Many of these fragments were too small or did not intersect to form a vertex and thus were not used in the initial cluster of assignments. We define the concept of edge coverage in Figure 8 which is defined as the percentage of predicted model boundary actually adjacent to image edge segments. The maximum edge coverage achievable in practice is on the order of 60%. This figure occurs for a well segmented model outline and for close model to image alignment.

The model edges and image edge segments are associated as follows:

- eliminate model edges occluded by the object itself;
- remove projected model edges shorted than 10 pixels;
- for each model edge collect all image edges with endpoints within 10 pixels of the projected model edge and orientation within .2 radians with respect to the projected model edge orientation.

4.1 Determining Edge Coverage using Distance Transforms

The method for determining edge coverage described above is essentially based on computational geometry. The drawback is that all of the edges in a segmentation must be considered when forming the correspondences with the projected model edges. In a large image this can mean that thousands of edges must be tried for



$$\% \text{ Edge Coverage} = \frac{\text{Total Segmentation Length}}{\text{Total Model Visible Occ. Length}}$$

Figure 8: Edge coverage is the fraction of predicted model projection boundary perimeter actually covered by extracted image edge segments.

each model edge. We have recently implemented a new method for determining edge coverage using a modified form of chamfer matching [Barrow *et al.*, 1977] which eliminates this problem.

4.1.1 Distance Transforms in Digital Images

A distance transform converts a binary digital image, consisting of feature and non-feature pixels, into an image where all non-feature pixels have a value corresponding to the distance to the nearest feature pixel. While in principle computing these distances is a global operation, there are algorithms that consider only small neighborhoods, but still give good approximations to the Euclidian distance. The algorithms are known as chamfering, because of the way in which the global distances are approximated by propagating local distances. Both sequential and parallel algorithms are well known.

In practice one starts with an image in which feature pixels are set to the value zero and all other pixels are set to infinity, i.e. a suitably large number. A given distance transform is characterized by a mask whose entries are the local distances that are propagated over the image. In the sequential algorithm, the mask is separated into two mask. The masks are passed over the image once each; the forward one from left to right, top to bottom and the backward one from right to left, bottom to top. The origin of the mask is placed over each pixel in the original image. The local distance in each mask entry is added to the value of the image pixel "under" it. The new value of the central pixel is the minimum of these sums. Borgefors [1986] considers many different masks and concludes that for most applications the "Chamfer 5-7-11", shown in Figure 9, mask provides the best approximation to the Euclidean distance.

-	11	-	11	-
11	7	5	7	11
-	5	0	5	-
11	7	5	7	11
-	11	-	11	-

Figure 9: The "Chamfer 5-7-11" distance transform mask.

4.1.2 Computing the Edge Coverage

In our application the features are the edges in the segmentation. Figure 10 shows the distance transform of a corner of a typical segmentation. The pixel values correspond to the distance to the nearest edge. It should be noted that the distance transform of the segmentation is computed only once for a given segmentation and then used for each pose to be confirmed. In addition, in our implementation because of efficiency considerations, distances greater than 20 pixels are not carried forward in the computation.

The edge coverage could be computed by projecting the model edges on to the image plane and traversing each edge pixel by pixel, keeping count of the total number of pixels and the number pixels that are within a certain distance, in our case 10 pixels, from a segmentation edge. (The distance of any given pixel to the nearest edge is given by the distance image.) The total edge coverage then, is the ratio of the latter quantity to the former quantity.

However, as noted by Barrow *et al.* [1977] position alone is not enough of a distinguishing feature. One can imagine that an edge passing through a sufficiently dense

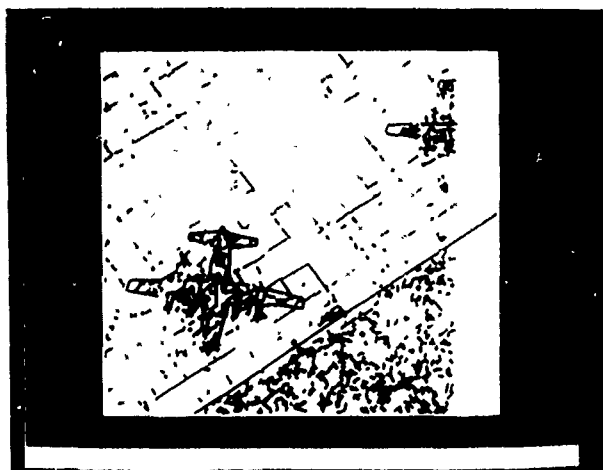


Figure 10: The distance transform of a typical segmentation.

area of the segmentation could accumulate a relatively high coverage figure even though it is not aligned with any of the segmentation edges.

To solve this problem, we allow each pixel of the image to become a list. Each entry in the list contains two pieces of information:

- 1) the distance to an edge in the image; and
- 2) the orientation of that edge.

The distance transform algorithm was modified to carry the orientation information along with the distance and to keep the information about edges with differing orientations distinct. Since information about edges greater than 30 pixels away is not carried forward in the calculation, the number of entries at each pixel is kept to manageable size. The final result is a distance image in which each pixel has information about the the distance and orientation of all edges within 20 pixels.

The edge coverage is then computed as described above except that to count a model edges pixel as good, it must not only be within 10 pixels of a segmentation edge but the orientations must be within .1 radians of each other. The combination of distance and orientation criteria have proven to be very robust.

In addition to being more reliable than the edge-grouping scheme for confirmation, this approach is considerably faster. Each pose takes less than one second to evaluate on a Symbolics 3600. With a typical large image on the order of 5×10^3 Hough space clusters are found, so the confirmation process takes less than two hours. The previous approach took between 20 and 30 hours to confirm the complete image.

5 Benchmark Test

The performance of the vertex-pair recognition algorithm is being evaluated in the context of an aerial reconnaissance application. A database of about 50



Figure 11: A typical benchmark image.

aerial photographs of a military airfield have been collected over a period of several months. The views are taken from a wide range of viewpoints and with variable weather conditions. This image database represents a realistic evaluation of our system to carry out routine airfield monitoring.

The images were collected on Kodak TMAX 100 35mm film and taken from an altitude of about 2000 feet. The spatial resolution of TMAX 100 is approximately 70 lines/mm. The central 24x24mm region of the film negative was scanned and digitized at a resolution of 2048x2048 with 256 grey levels. The main target for recognition is the C130 transport aircraft which subtends a range of 150 to 250 pixels in the digitized images. A typical airfield view is shown in Figure 11. The resulting image segmentation is shown in Figure 12. The segmentations are produced by a modified form of the Canny edge detector followed by line segmentation based on chain curvature extrema [Canny, 1983, Asada and Brady, 1984]. A peak edge detector is also employed to remove edges corresponding to thin lines which cannot correspond to model projection edges.

To date, we have processed 5 of these airfield views which are blocked into 600x600 pixel tiles, for a total of about 250 tiles. The tile blocking allows distributed processing over a number of Symbolics 36xx machines. The tiles are defined with overlap so that any instances of a target are complete in at least one image tile. Tiles with fewer than 10 edges are ignored so that a typical 2048x2048 image yields 40-50 tiles.

Two examples of the polyhedral models used in the experiments are shown in Figure 13. The building model is used to establish the overall camera transformation for the airfield. A sample recognition result is shown in Figure 14. The models are shown aligned with the image according to the transformation clusters with best edge coverage ratio. (The actual edge coverage value is shown next to each model.) An alternative view of the site configuration is shown in Figure 15 to illustrate that

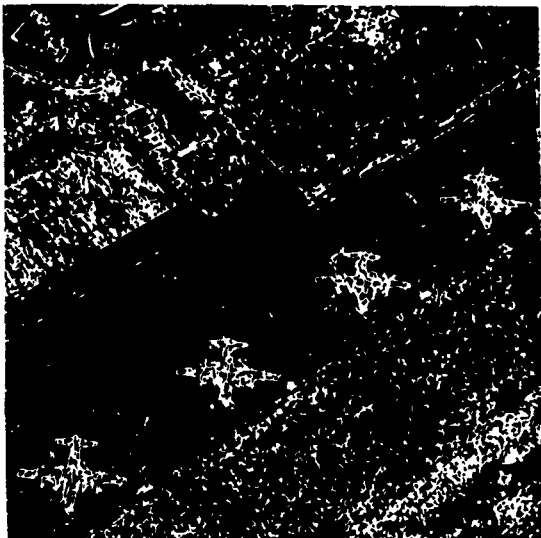


Figure 12: The segmentation of the image in the previous figure.

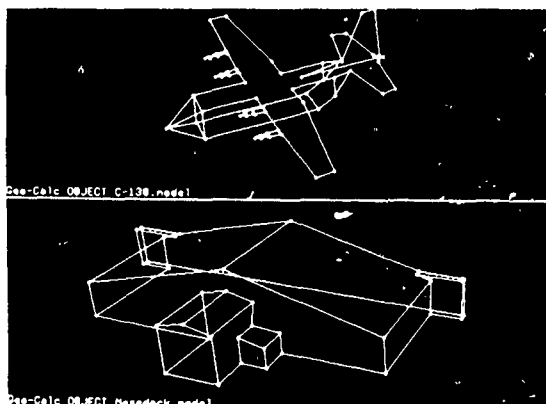


Figure 13: Polyhedral models for the C130(top) and an airfield hangar(bottom).

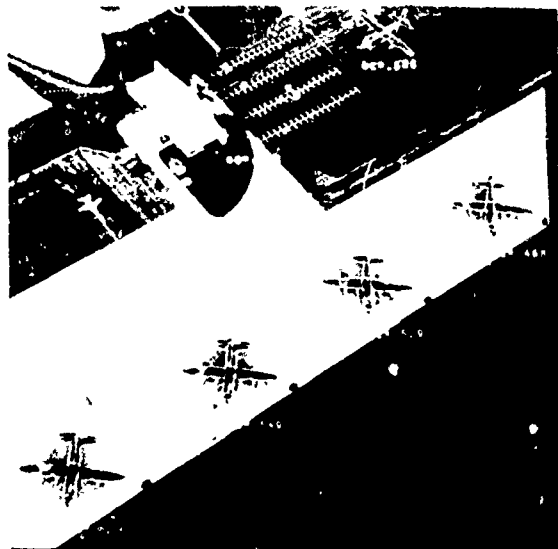


Figure 14: A sample recognition result.

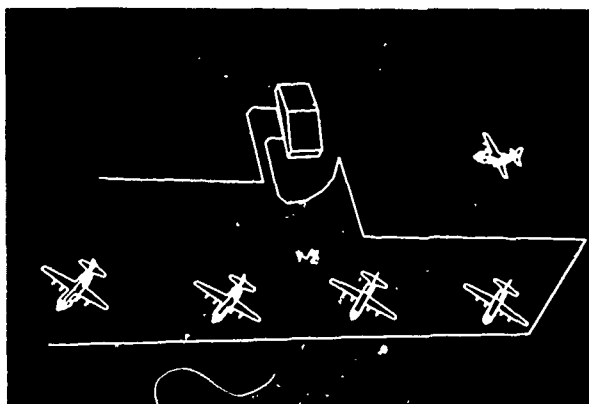


Figure 15: An alternative view of the three-dimensional model configuration produced by the vertex-pair recognition algorithm.

the recognition process produces a full three-dimensional representation of the airfield.

6 Empirical Results

6.1 Computational Requirements

Table 1 shows the numbers of the various data structures used by the vertex-pair recognition algorithm to process the image shown in Figure 11. There is roughly a thousand-fold reduction in the data from image to valid-clusters.

6.2 Recognition Results

Five images containing 30 instances of C-130 aircraft were divided into a total of 250 tiles and processed. Of these 250 tiles, two were misclassified. One contained an aircraft which was not found and one contained a collection of random image features that was classified

STRUCTURE	# INSTANCES
Model Vertex-Pairs	31
Pixels	4.2×10^6
Edges	1.6×10^4
Vertices	1.7×10^4
Segmentation Vertex-Pairs	3.6×10^6
Assignments	6.3×10^7
Hough Clusters ($n_a \geq 4$)	5.9×10^3

Table 1: The number of instances of various structures used by the matcher.

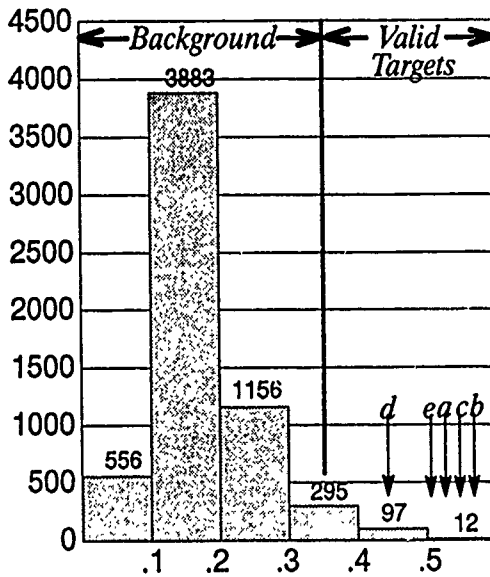


Figure 17: The distribution of edge coverage values.

as a C-130. Both errors were traced to problems in the digitization or segmentation process.

6.3 Effects of Cluster Size

During the initial parameter adjustment experiments in preparation for the benchmark, we found that even with 31 model vertex-pairs we had to consider all clusters with four or more assignments to assure robust operation. In order to study the discrimination power of clustering alone, we reanalyzed the match solutions varying the minimum acceptable cluster size. The results for our example image are shown in Figure 16.

6.4 Effect of Edge Coverage

Figure 17 shows the distribution of edge coverage values computed by the hypothesis confirmation algorithm for the sample image. Values greater than $\frac{1}{3}$ have been found to reliably correspond to valid instances C-130 aircraft in the images in the test database.

6.5 Effect of Number of model features

The automatic vertex-pair selection algorithm picks in the the range of 20 to 35 vertex pairs to characterize the models which we use for matching. In early experiments with hand-picked model vertex-pairs we would typically use four to six pairs to characterize a model, so we initially suspected that the automatic algorithm, in choosing five times that number, was picking a somewhat sub-optimal set. However, as mentioned earlier, it was found that at least 31 model vertex-pairs were needed for reliable operation on all of the images in the test suite.

To determine the effect on recognition performance, the match solutions were reanalyzed by removing model vertex-pairs from the solution clusters and discarding those that fell below the minimum size of 4 assignments. The error measure was used to order the vertex-pairs from for removal. Those with poor performance were removed first. Figure 18 shows the results. The peak discrimination, in this instance, occurs at 10 to 12 model vertex-pairs. This is typical number, although the individual vertex-pairs comprising this set varies from image to image.

The need for a much larger set of model vertex-pairs arises from the effect where by the geometry of a feature in the segmentation is dependent on the size of the given feature in the image. For example, at small scales the wing tip of the C130 show up as a single edge, whereas at large scales it is broken up and may appear as two or more edges in the segmentation. The current system has no way to represent these scale-dependent effects and consequently additional model vertex-pairs are needed. Work is currently underway to characterize these effects for inclusion in future versions of object recognition systems.

6.6 Tracking Assignments Through Hough Space Clustering Stages

The discrimination power of the Hough space clustering technique arises from the size of the six-dimensional space ($> 10^{12}$ bins). This also presents a problem to the implementor in that the entire space cannot be represented in a computer at once. As explained earlier, this is addressed by factoring it into smaller sub-spaces, in this implementation $[\phi, \psi], [\zeta], [\mu, \nu], [w^{-1}]$.

Another implementation decision was that since ϕ and ψ were the first and cheapest (a table lookup) transform components to be found, it made sense to cluster on these parameters first. Early matching experiments were conducted on small images with the object of interest occupying a significant fraction of the image area. In addition, only four to six model vertex-pairs were employed.

These experiments indicated that a significant number of assignments were eliminated at this stage and leaving very little work for later clustering stages. The original paper on the vertex-pair matching algorithm [Thompson and Mundy, 1987b] asserted that this was a significant feature of the algorithm.

A recently published theoretical analysis of the use of the Hough transform in object recognition [Grimson

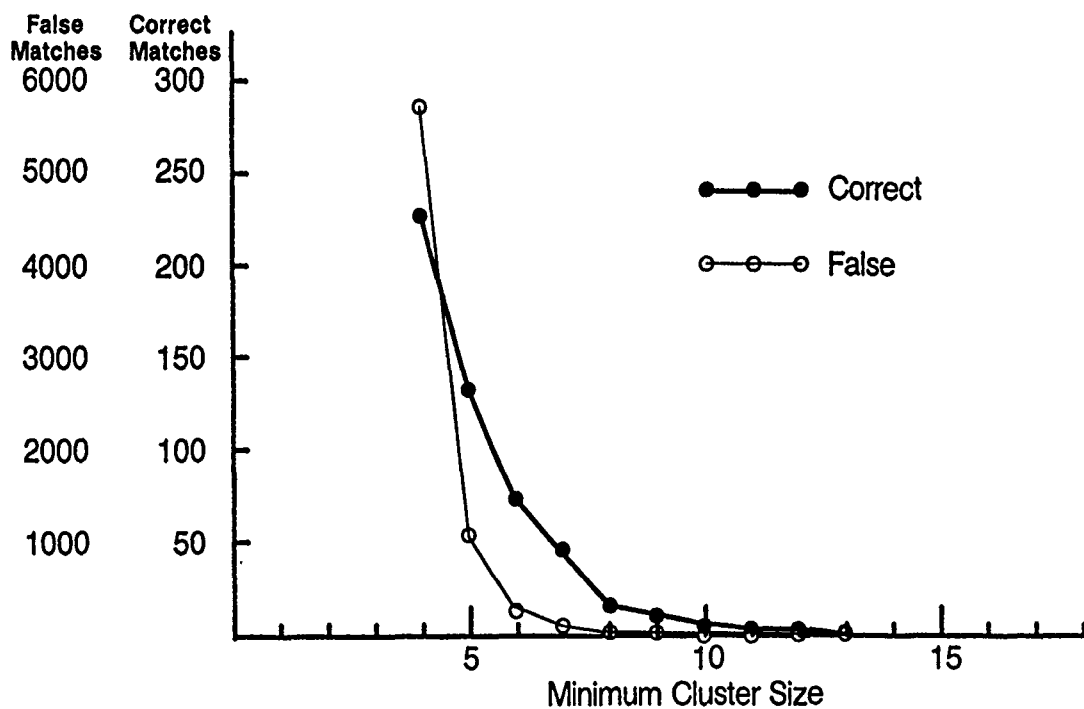


Figure 16: The effect of minimum cluster size (31 model vertex-pairs).

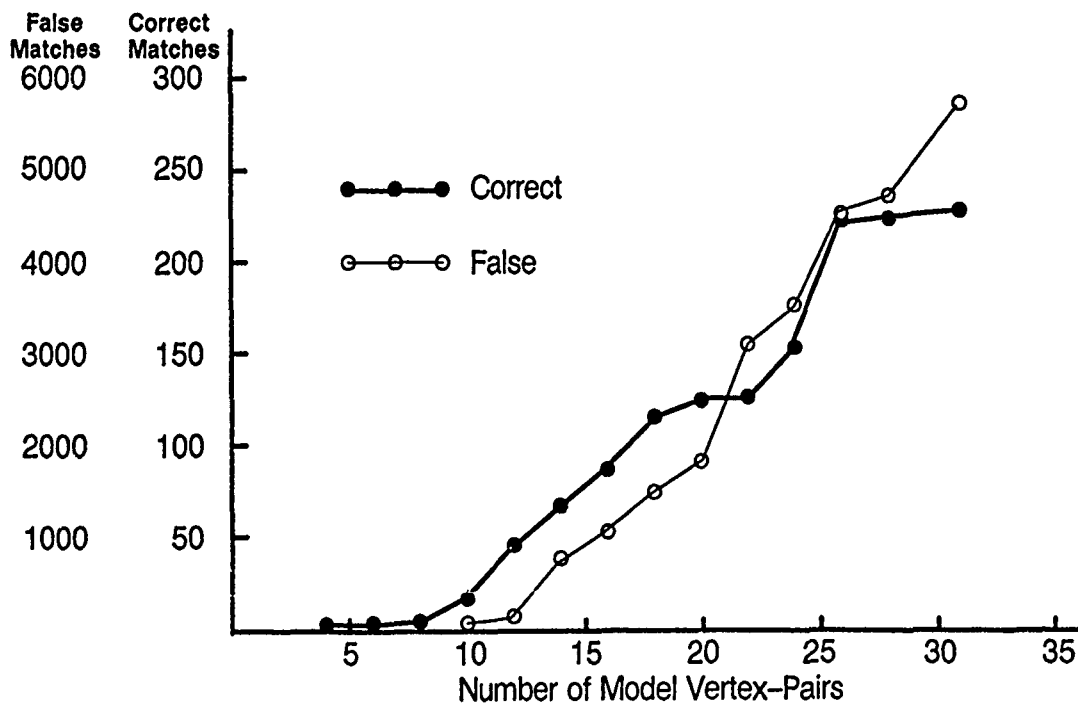


Figure 18: The effect of the number of model vertex-pairs (minimum cluster size four).

and Huttenlocher, 1990] refutes this and goes on to claim that for most images, virtually no assignments are ruled out by the clustering on rotation alone and that, in fact, many incorrect assignments survive clustering in the full six-dimensional Hough space.

We have been in agreement with the latter assertion for sometime, and in fact, this has been the among motivations for studying the error performance of model vertex-pairs and techniques for hypothesis confirmation. However, we should reiterate that Hough space clustering still provides over a thousand-fold reduction in the data, yielding a relatively small number of poses that can be rapidly assessed by other means.

To test the former assertion, we added code which tracks the individual assignments throughout the clustering process. With this arrangement, we examined images yielding in the range of 3×10^3 to 2×10^5 assignments. For these images, 2 to 10 percent (with a mean of 5 percent) of the assignments were filtered out the rotational clustering stages. There was no significant correlation between the total number of assignments and the percent remaining. While these figures are not quite as small as the theoretical prediction ($\leq 1\%$), they are small enough that we now agree that a significant amount of work remains to be performed by the clustering on the remaining three transform parameters.

7 Conclusions

The experiments have resulted in less than 1% of the tiles misclassified. There was one false positive and one false negative classification in over 250 tiles. The false positive target indication was produced in the first image of the test series and before we introduced the removal of peak edge contours. The false negative classification was due to very poor image exposure which can easily be improved.

We were not able to handle images with heavy cloud cover such as the case shown in Figure 19. In these images, the image segmentation process is not successful in recovering any significant portion of the target boundaries. Images of this type were eliminated from the benchmark test, since a completely different approach to image segmentation and object representation would be needed to handle such cases.

The overall conclusion is that the current system can be used in a routine airfield monitoring application where the image contrast and resolution are sufficient to produce reasonable segmentation of the target boundaries.

References

- [Asada and Brady, 1984] H. Asada and M. Brady. The curvature primal sketch. In *Proc. IEEE Workshop on Computer Vision: Representation and Control*, 1984.
- [Ayache and Faugeras, 1984] N. Ayache and O. Faugeras. A new method for the recognition and positioning of 2d objects. In *Proc. Seventh Int. Conf. on Pattern Recognition*, pages 1274-1280, 1984.
- [Ballard, 1981] D. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13:111-122, 1981.
- [Barrow et al., 1977] H. G. Barrow, J. M. Tannenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proc. Fifth Int. Joint Conf. on Artif. Intell.*, pages 659-663, Cambridge, MA, 1977.
- [Borgefors, 1986] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34:344-371, 1986.
- [Canny, 1983] J. Canny. Finding edges and lines in images. Report AI-TR-720, MIT, Artificial Intelligence Laboratory, 1983.
- [Grimson and Huttenlocher, 1990] W. E. L. Grimson and D. P. Huttenlocher. On the sensitivity of the hough transform for object recognition. *IEEE Trans. Pattern Anal. and Machine Intell.*, PAMI-12(3):255-274, 1990.
- [Grimson and Lozano-Perez, 1985] W. E. L. Grimson and T. Lozano-Perez. Search and sensing strategies for recognition and localization of two and three dimensional objects. In *Proc. Third Int. Symposium on Robotics Research*. MIT Press, 1985.
- [Lowe, 1985] D. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, Boston, MA, 1985.
- [Roberts, 1965] L. G. Roberts. Machine perception of three-dimensional solids. In J. T. Tippett et al., editor, *Optical and Electro-Optical Information Processing*, pages 159-197. MIT Press, 1965.
- [Silberberg et al., 1986] T. Silberberg, D. Harwood, and L. Davis. Object recognition using oriented model points. *Computer Vision, Graphics and Image Processing*, 35, 1986.



Figure 19: An image with heavy cloud cover. Images of this type cannot be analyzed by the current system.

- [Slama, 1980] C. C. Slama, editor. *Manual of Photogrammetry*. American Society of Photogrammetry, Falls Church, VA, fourth edition, 1980.
- [Thompson and Mundy, 1987a] D. W. Thompson and J. L. Mundy. Model-directed object recognition on the connection machine. In *Proc. DARPA Image Understanding Workshop*, 1987.
- [Thompson and Mundy, 1987b] D. W. Thompson and J. L. Mundy. Three-dimensional model matching from an unconstrained viewpoint. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 208-220, 1987.
- [Whitney, 1955] Whitney. On singularities of mappings of euclidean spaces; i. mappings of the plane onto the plane. *Annals of Mathematics*, 62, Nov. 1955.

A MULTISTRATEGY LEARNING APPROACH FOR TARGET MODEL RECOGNITION, ACQUISITION, AND REFINEMENT

John Ming and Bir Bhanu

Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, Minnesota 55418

ABSTRACT

Target recognition systems are currently unable to modify their behavior automatically in environments where processing requirements change or novel situations are encountered. Most systems can not easily adapt to varying target appearances, considerable image noise, and target occlusion. More importantly, these systems are constrained by the selection of target models used for recognition; typically, the target model database is fixed and individual features within a target model remain static as well. The incorporation of machine learning technology into the target recognition process will allow the system to use situation context, to adapt in changing environments, and to improve the system's performance over time. This work describes an innovative approach which combines machine learning and target recognition into an integrated system. The system is called **TRIPLE: Target Recognition Incorporating Positive Learning Expertise**. It uses two machine learning techniques known as explanation-based learning and structured conceptual clustering, combined in a synergistic manner, which provide effective target model recognition, acquisition, and refinement capabilities. We provide an overview of the TRIPLE system and provide experimental results which illustrate the performance of the system.

1. INTRODUCTION

Prior attempts to automate target recognition systems have suffered from the lack of an ability to automatically acquire new target models, to adapt to changing environmental conditions, and to modify system behavior based on the context of the situation in which the systems are operating. In order to be effective in dynamic outdoor scenarios, a robust vision system should be able to automatically acquire necessary contextual information from the environment and react accordingly. Most target recognition systems lack this capability. Their performance begins to quickly degrade when subjected to problems such as variable target appearance, image noise, and target occlusion.

Due to recent advances in machine learning technology, some of these problems are resolvable by effectively combining machine learning and machine vision technologies. Learning allows an intelligent vision system to use situation context in order to understand images. This context, as defined in a machine learning scenario, consists of a collected body of background knowledge as well as environmental observations which may impact the processing of the scene. The resulting system dynamically reacts to the appropriate stimuli in the environment, continuously adapting its internal

knowledge to improve overall performance levels. This improvement may come in the form of faster recognition times, improved recognition accuracy, and higher confidence in system results.

Machine learning technology should facilitate two main advancements in the target recognition domain: automatic knowledge base acquisition and continuous knowledge base refinement. The use of learning in the construction of the knowledge base will save the user from spending the enormous amount of time necessary to derive target models and databases. Knowledge base refinement can then be incorporated to make any necessary changes in the system's database to improve the performance of the vision system. These two modifications, which are the focus of this paper, will serve to significantly advance the state-of-the-art in target recognition and image understanding applications.

Although machine learning has been used in many applications, very little work has been done in the computer vision domain. Refer to the earlier review by Bhanu and Ming² for an overview of computer vision and machine learning systems. Further, within the machine learning field, very little effort has been made to combine several learning techniques together. Typically, learning methodologies are used independently to provide adaptive ability and improved system performance. Our multistrategy approach to target recognition presented in this paper, called **TRIPLE** (Target Recognition Incorporating Positive Learning Expertise), incorporates two powerful learning techniques, known as explanation-based learning (EBL) and structured conceptual clustering (SCC). These techniques filter and structure the information present in positive concept examples to create useful knowledge structures. While each of these learning methods, used independently, might provide some improvement in target recognition performance, they can be best put to use by combining their abilities into an integrated approach. We have synergistically combined the EBL and SCC learning methodologies in the TRIPLE system to offer a consolidated technique which employs the best features of each method to solve the target recognition problem in an efficient and effective manner.

2. TRIPLE - A MULTISTRATEGY LEARNING TECHNIQUE

The TRIPLE target recognition system integrates the EBL and SCC learning techniques to overcome the inherent limitations present in each approach. EBL,^{3,6} which is classified as a *learning by observation* technique, uses inference to construct a useful concept description from a single example of that concept. Derived from earlier learning systems which required a large number of examples in order to

generalize a target description, EBL uses a collection of applicable background knowledge to generate a useful target description from a single example. EBL's main limitation is the recognition time required when the number of target models becomes large. SCC^{4,5,7} is a method for grouping targets into classes similar to traditional numerical clustering techniques. However, instead of using predefined measures of target similarity to determine class boundaries, SCC uses a conjunction of conceptual attributes to group targets into conceptually simple classes. This process utilizes important contextual information relevant to the targets to assist in the classification process. SCC can handle complex, structural descriptions of targets, which is ideal for target recognition tasks since most targets are represented using structural descriptions. However, SCC has problems with model biases when the number of target class examples is small. Combining the ability of EBL to characterize a target using a single training example with SCC's efficient method of organizing targets once they have been properly modeled yields an integrated learning system which effectively handles the target recognition task. A more complete description of the EBL and SCC learning approaches is presented in our earlier description of the TRIPLE system.²

Figure 1 shows the current configuration of the components in the TRIPLE target recognition system. The processing elements, which are indicated by rectangular boxes, transform the input image data and generate the target recognition results. The Segmentation and Symbolic Feature Extraction component segments and locates the regions of interest in the original image and then extracts the symbolic feature information from these regions. The Knowledge-Based Matching component parses the classification tree using the symbolic target features and identifies the various recognition states of the TRIPLE system. The matching component also initiates the proper learning cycle based on the target recognition results. EBL, when invoked by the matching component, selects the relevant target features from the symbolic feature information during the target model acquisition process. EBL also identifies new, pertinent target features for target models already present in the classification tree. SCC is responsible for constructing and maintaining the target classification tree using the relevant symbolic features selected by the EBL component. The Feature Value Monitor modifies the target feature values in the classification tree based on the features which are used to identify the target during the recognition cycle.

The processing elements make use of several collections of target-specific data and knowledge databases within TRIPLE, as shown in Figure 1. The image data is assumed to contain targets of interest and may include targets that are not currently in the target model database. The Symbolic Feature Definitions are used during the symbolic feature extraction process to identify important target features which are used to recognize the target. The Background Knowledge is accessed by the EBL component to select relevant target features during the model acquisition or model refinement operations. The Target Model Database stores the complete schema of each target encountered by the recognition system, including every feature (relevant or not) defined on each target. Relevant target features, which are determined by the EBL component, are tagged for future reference in the target model. SCC makes use of the Goal Dependency Network while constructing or modifying the classification tree in order to compute the optimal clustering of the targets at the current level in the target hierarchy. The Target Classification Tree represents a structured hierarchy of all targets known by the TRIPLE system and is used by the matching component to identify various types of target recognition results such as complete recognition, partial recognition,

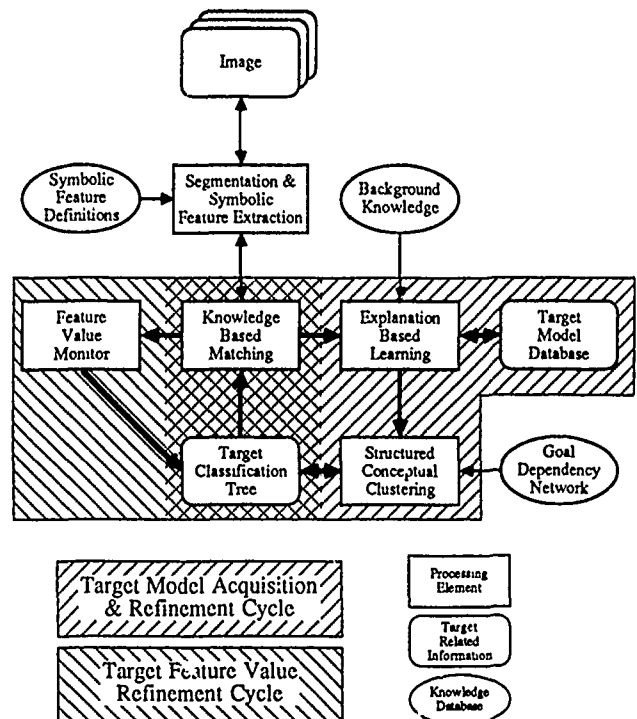


Figure 1: Multistrategy Machine Learning Approach for Target Recognition.

occlusion, new target, target model refinement, or recognition failure.

Figure 1 also highlights the two distinct learning cycles which are present in the TRIPLE system. The first learning cycle is the target model acquisition and refinement process. The components used in this loop include the knowledge-based matching, explanation-based learning, and structured conceptual clustering processing elements as well as the target model database and the target classification tree. This learning cycle also includes the background knowledge base and the goal dependency network associated with the EBL and SCC learning components, respectively. The second learning cycle within TRIPLE is the target feature value refinement process. This operation utilizes the knowledge-based matching and feature value monitor components. The feature value monitor modifies the relevant target feature values that were used by the knowledge-based matching component to recognize the target and which are present in the target classification tree.

The various processing components of the TRIPLE system and the manner in which they interact will now be described in the following subsections.

2.1 Segmentation and Symbolic Feature Extraction

The first step in the target recognition process is to extract sufficient information from an image so that targets present in the image can be correctly identified. This process is handled by the Segmentation and Symbolic Feature Extraction component. First, the image is segmented using a set of selected parameters for a given segmentation algorithm. From the segmentation results, the target-related regions are identified and approximated using piecewise linear segments. Based on the size, shape, and relationships with neighboring regions of the border approximation, a region label is assigned to each target segment. A hypothesize-and-test approach is used to identify the rough orientation of the tar-

get during the region labeling operation.

For example, in the aircraft recognition scenario, the feature extraction process may first hypothesize a region in the image that corresponds to an aircraft fuselage based on its shape properties (narrow, elongated region). This hypothesis can then be verified by finding a symmetric pair of regions adjacent to the fuselage with wing-like properties. Similarly, the tail regions of the aircraft can also be labeled and used to support the current hypothesis.

When a hypothesis has been verified using surrounding regions as additional evidence, all contributing regions are tagged and used in the symbolic feature extraction operation. TRIPLE computes symbolic feature information from the region borders using a knowledge-based approach. *Symbolic features* represent conceptual descriptions of a target's properties that would be used by a human in characterizing the target's appearance. Each symbolic feature is represented by a set of production rules that are stored in the Symbolic Feature Definitions database in Figure 1. The conditions of the rule analyze the properties of the target regions and the actions define the appropriate symbolic feature when the conditions are satisfied. The rules use distances and orientations of line segments on the region borders to compute the various target features. For instance, in the aircraft recognition example, symbolic features such as fuselage length, wing span, and wing sweep angles can be obtained by examining the fuselage and wing regions of the target.

2.2 Knowledge-Based Matching

The knowledge-based matching component receives a target schema from the segmentation and symbolic feature extraction component, which represents the feature information obtained for the unknown target. This schema is utilized by the matching element to traverse the classification tree in an attempt to reach a leaf node of the tree. If successful, the target has been correctly identified. The classification tree represents a structured hierarchy of the target models currently known by the recognition system. The tree is constructed and maintained by the SCC component (Section 2.4). The use of the classification tree makes the target recognition process much more efficient and effectively solves the "indexing" problem encountered in target recognition applications. And, as we shall see later, the classification tree makes it possible to identify situations such as target occlusion or incomplete target recognition that would not otherwise be possible.

The matching process begins at the root of the classification tree, matching the feature values specified in each tree branch with those in the unknown target schema. Finally, when a leaf node is reached, the matching operation terminates by matching any remaining feature values. If at any point in the tree traversal, a feature is missing from the unknown target schema, the system spawns a set of multiple *viewpoints*. A separate viewpoint is created for each feasible branch at the current level in the classification tree. This action allows the tree parsing process to evaluate many hypothetical alternatives. The survival of any given viewpoint is governed by the matching success that is achieved during the processing of successive tree nodes in that viewpoint. At a later time in a particular viewpoint, the tree parsing process may terminate due to feature incompatibility. This condition results in the removal of the corresponding viewpoint from further consideration. Viewpoint removal allows the search process to prune branches from the classification tree when it becomes clear further search will be useless. If a path through the entire tree to a single leaf node can be located, the unknown target

has been correctly recognized as the target model present at the leaf node.

If the knowledge-based matching component is unable to parse the tree using the available symbolic feature data, the feature set is passed to the EBL component. In these situations, the failure of the matching process is due to one of two conditions. First, the feature information may represent a new target model that is not currently represented in the classification tree and which can potentially be acquired by the EBL component. Alternatively, the feature data may be faulty, incomplete, or inconsistent with the system's current target recognition domain, in which case EBL will not be able to acquire a new model. However, the matching process does not distinguish between these two cases. It merely passes the feature information to the EBL component for further investigation.

In addition, the matching component also sends the feature information to the EBL component when it detects the presence of a new feature in a correctly recognized target. By consulting the information stored in the Target Model Database, the matching component can detect when a new feature is present. It adds the new feature to the current target model and passes the revised model to the EBL component in order to determine the relevance of the new feature.

2.3 Explanation-Based Learning

Whenever the knowledge-base matching component is unable to process the symbolic feature information for an unknown target, the Explanation-Based Learning (EBL) component is invoked to understand the feature data. Since it makes use of a collection of domain-specific Background Knowledge (Figure 1), EBL is able to draw inferences from the symbolic feature data that are not possible for the matching process using the classification tree alone. But, because EBL and its associated knowledge base are only used in situations where the classification tree fails, the TRIPLE system remains highly efficient by accessing the information in the knowledge base only when necessary. Complete details of the EBL process are provided by Bhanu and Ming.²

The EBL component is responsible for four separate tasks within the TRIPLE target recognition system:

- (1) *Processing the training examples during system initialization.* The TRIPLE system uses EBL to simplify the target modeling process, which has traditionally been very difficult due to the amount of work necessary to generate a correct model. EBL applies the background knowledge base (Figure 1) to each target schema using a generic target prototype to guide the explanation process. Once the explanation has been created, it is generalized to create a target model that contains the relevant target features. This model can then be used to recognize subsequent instances of the target. All target models created during system initialization are sent to the SCC component, which generates the target classification tree.
- (2) *Acquiring new target models.* When the target classification tree is unable to process an unknown target schema, the EBL component is given the feature data in order to determine if a new target model can be constructed from the available features. The new model acquisition process is identical to the system training process described above. If a new model can be successfully derived, it is added to the target model database and is passed on to SCC for addition into the current target classification tree.

- (3) *Refining existing target models.* EBL is also invoked to determine the relevance of new feature information that is present for an existing, correctly recognized target model. The presence of a new feature can be detected since EBL maintains a list of all previous symbolic features defined on each target in the target model database (Figure 1). EBL adds the new feature to the current target model feature set and reprocesses the feature data. If the new feature is found to be relevant, it is tagged in the target model and is sent to the SCC component for addition into the target classification tree. Otherwise, the feature is simply left in the target model database as non-relevant.
- (4) *Identifying recognition failures in the TRIPLE system.* EBL is responsible for determining cases of recognition failure. When the knowledge-based matching component is unable to process a set of feature data, EBL is given the chance to acquire a new model using the available features. However, if EBL can not construct an appropriate model from the feature information, the feature set is incomplete or the background knowledge is insufficient to understand the feature data. In either case, the situation is reported as a recognition failure.

2.4 Structured Conceptual Clustering

The Structured Conceptual Clustering (SCC) component of the TRIPLE system constructs the Target Classification Tree from the relevant feature data generated by EBL (see Figure 1). The classification tree represents a structured hierarchy of the targets currently stored in the recognition system. As described earlier, traversal of the classification tree allows the matching component to understand and compensate for missing information in unknown target schemata during the recognition process. The classification tree also provides efficiency in the target recognition task since the matching process does not have to compare the unknown target schema with every target model currently in the target model database (i.e., the indexing problem is effectively handled). Thus, the SCC component plays a vital role in the TRIPLE system since it generates and maintains the structure of the classification tree.

During the construction or modification of the target classification tree, SCC accesses the information present in the Goal Dependency Network (GDN) in order to select useful target features. Global target characteristics are specified by the GDN at high levels (near the root) in the tree because they usually categorize coarse target classes. Within these classes, the GDN suggests more specialized target features that are used to determine subclass assignments. This approach to tree construction also allows the matching component to make appropriate decisions (complete recognition, incomplete recognition, target occlusion, etc) during the target recognition process.

Although the GDN suggests several features to use at a particular position in the tree, the SCC process must still select the best feature for the specific situation. To perform this task, each suggested feature is used to generate a clustering of the targets. The quality of each clustering is based on the *conceptual simplicity* of the clustering results. TRIPLE uses several factors in determining the conceptual simplicity of a proposed clustering including: the number of clusters into which the targets have been placed; the inter-cluster and intra-cluster distances of the clustering results; and the GDN's ranking of the selected feature at the current level in the tree. These measures of clustering quality are combined to evaluate the clustering results of each feature. The feature that provides the highest clustering quality value is selected

and the current branch of the tree is defined accordingly. The SCC component continues to cluster the targets at each branch in the tree until every single target has been placed into a separate leaf node in the tree.

SCC provides an adaptive capability to the TRIPLE system since it never relies on predefined measures of class similarity, but rather, it computes the feature that best distinguishes a set of targets at any given level in the classification tree. Over time, the choice of the distinguishing feature at a particular level may dynamically change as a result of the new targets and revised targets which are continually being placed in the classification tree. An analysis of the tree structure across many successive recognition cycles of the TRIPLE system shows that it dynamically responds to the targets which are added or modified by automatically restructuring the appropriate tree branches to obtain an optimal target categorization.

The SCC component performs three different jobs within the TRIPLE target recognition system:

- (1) *Construction of the initial target classification tree during system training.* SCC takes all the target models created by the EBL component and constructs the classification tree. At each branch in the tree, the GDN is used to suggest a set of appropriate target features from which one is selected by measuring the conceptual simplicity.
- (2) *Addition of a new target model into the target classification tree during the target model acquisition process.* SCC attempts to retain as much of the original structure of the tree as possible. SCC traverses the tree using the new target model until a branch is encountered that is not compatible with the new target's features. The tree is then reclustered at that location. If a leaf node is encountered, a new branch is created to distinguish the target model currently stored in the leaf node from the new target model.
- (3) *Modification of the current OCT structure during the target model refinement process.* This process is similar to the new target model situation since SCC minimizes the required changes to the tree. At each node in the tree, SCC determines if the new feature produces a better clustering quality than the distinguishing feature used at the current branch. If the new feature is better, the tree is reclustered at the current location. Otherwise, the appropriate branch is selected and the process continues. If a leaf node is reached, the new target model feature is simply inserted at the leaf node.

2.5 Feature Value Monitor

The Feature Value Monitor updates the quantitative feature values of a target model, if and when that model is used to recognize an unknown target. This process allows the TRIPLE system to gradually modify the feature values of a target in order to overcome any initial bias that may have been acquired during the initial construction of the target model. Changes in target models made by the feature value monitor will be very gradual compared with the changes which result from activating the EBL-SCC target model refinement process described earlier. In the latter case, symbolic features are added or removed from the relevant feature list of the target model. The feature value monitor simply modifies the *relevant* quantitative feature values of the target. Further, the feature value monitor does not modify any qualitative target features present in the model.

The feature values are changed by shifting the range of

numeric values produced during the EBL generalization process in the direction of the new target feature value. Each range is characterized by a central feature value with endpoints a prescribed distance away from this value. For example, the central feature value for the range (100'-196') is 103'. The feature value monitor moves the entire feature value range in the direction that more closely aligns the central value of range with the new target feature value. The width of the feature value range remains the same. To avoid wild fluctuations in the feature values ranges, the range is moved only one unit (one foot, one degree, etc) during any given recognition cycle, regardless of the discrepancy size. This approach is preferable to the alternative method of aligning the numeric range on the current target's feature value because it prevents potential misclassification results from adversely affecting the actual location of the feature value range. The approach is also more in tune with the notion that adaptation should be a gradual, rather than abrupt, process.

2.6 Recognition and Learning in TRIPLE

During every recognition cycle, the TRIPLE system identifies one of the following *recognition states*:

- (1) *Complete Recognition* - The unknown target schema is correctly classified with a high degree of confidence using the classification tree. The knowledge-based matching component and the feature value monitor are involved in the complete recognition operation.
- (2) *Incomplete Matching* - The unknown target schema is partially classified using the classification tree. The matching component identifies multiple target models in the classification tree which meet the limited constraints imposed by the available unknown target features. A recognition confidence is produced for each matched target model. Only the knowledge-based matching component is used in this operation.
- (3) *Target Occlusion* - Although occluded, the identity of the unknown target schema is predicted with some confidence level using the classification tree. This operation involves only the knowledge-based matching component.
- (4) *Target Model Acquisition* - The unknown target schema can not be classified using the current classification tree, so the target model is acquired by the EBL-SCC learning cycle and added to the classification tree. The model acquisition process involves the knowledge-based matching, EBL, and SCC components of the TRIPLE system.
- (5) *Target Model Refinement* - After correctly classifying the unknown target schema using the classification tree, a new feature is identified in the unknown target schema. The target model and the classification tree are updated to indicate the relevance of this new target feature. The model acquisition process involves the knowledge-based matching, EBL, and SCC components of the TRIPLE system.
- (6) *Recognition Failure* - The unknown target schema can not be classified using the information in the classification tree or by EBL with the use of the background knowledge database.

Figure 2 summarizes, in a decision diagram format, the conditions which lead to each of the six recognition states.

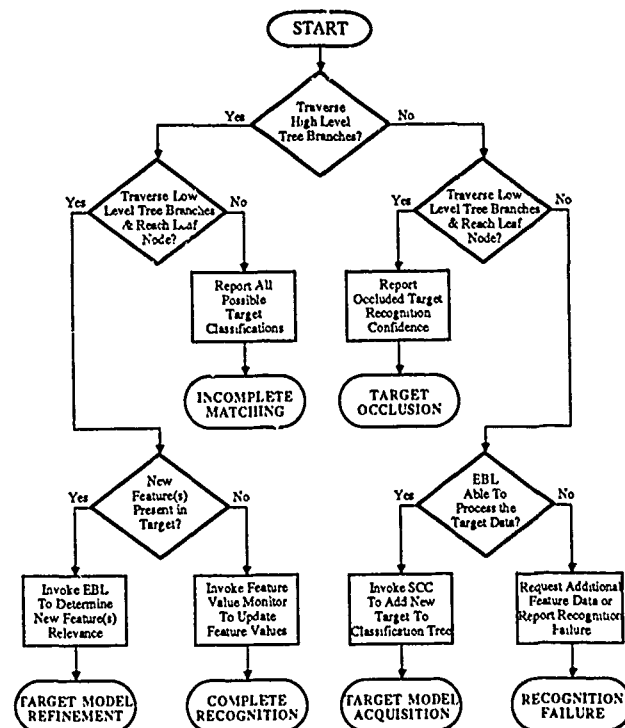


Figure 2: Decision diagram specifying the conditions through which the six different *recognition states* of the TRIPLE system are identified.

3. EXPERIMENTAL RESULTS

We have conducted a series of experiments to test the target recognition and learning capabilities of the TRIPLE system for the recognition of 2D aircraft. The imagery used for these experiments was generated by digitizing technical diagrams of various commercial aircraft ranging in size from small single engine private aircraft (Cessna Caravan) to large passenger airliners (Boeing 747). Eleven aircraft were selected for the initial set of experiments on the TRIPLE system.

Since the technical diagrams for the aircraft are extremely precise, they do not represent the actual appearance of aircraft seen in real imagery. In order to simulate the degraded appearance of the aircraft for our experiments, we introduced noise and distortion into the border approximations for the aircraft. Gaussian noise (mean = 0, variance = 1-20) was added to each of the border points and the resulting image was then distorted using two morphological operators (erosion and dilation). Once the aircraft image was distorted, a border following routine was invoked to generate the list of pixels that comprise the outline of the aircraft. The aircraft border was then represented with a piecewise polygonal approximation using a split-merge approximation algorithm. Figure 3 provides an example of the border distortion process and the corresponding polygonal approximation results for a typical aircraft.

The polygonal approximation for an aircraft is processed by a knowledge-based algorithm to create the list of symbolic target features needed by the TRIPLE system. Given the orientation of aircraft, the knowledge-based process analyzes the line segments in the polygonal approximation to derive various symbolic features. This operation makes use of the symmetry properties of the aircraft's shape in determining

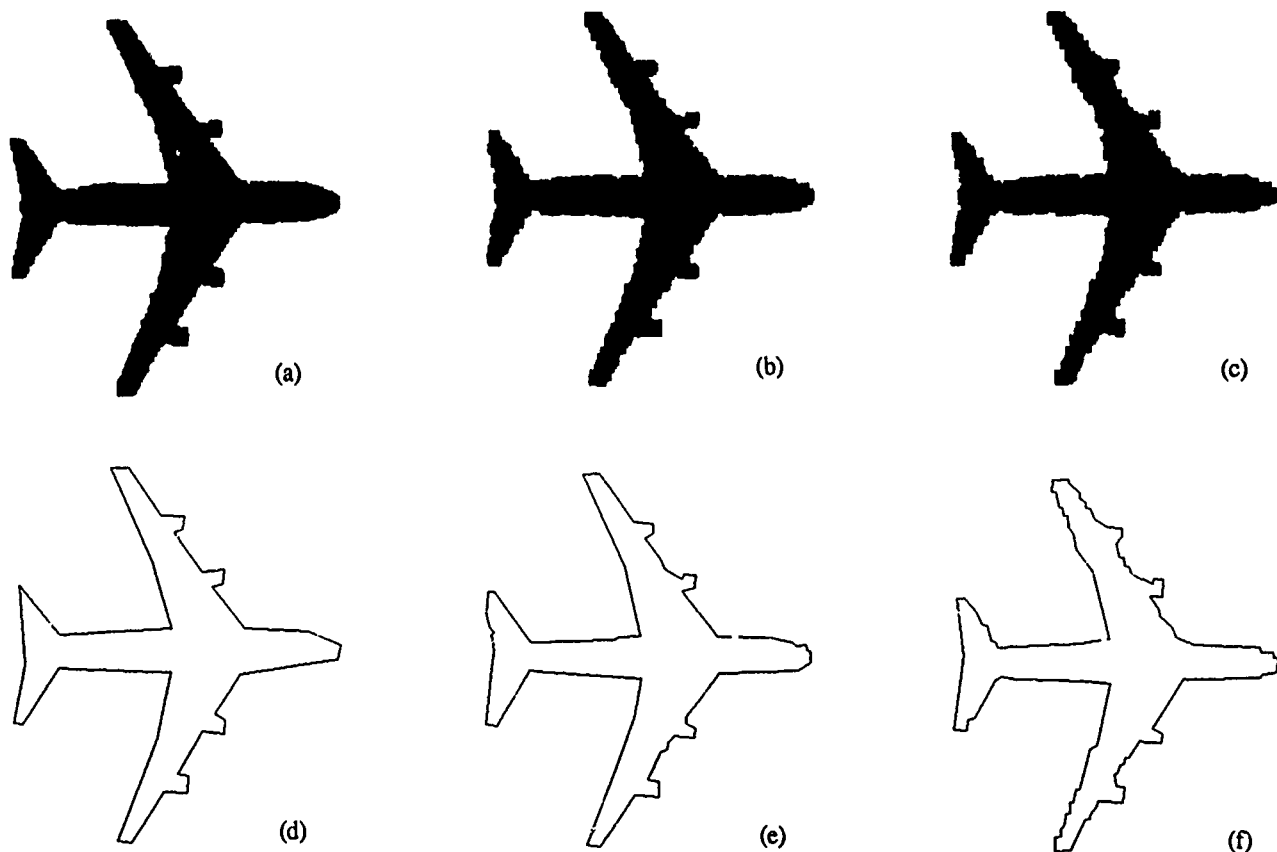


Figure 3: Various levels of noise and distortion added to an aircraft. (a) Noise level = 3. (b) Noise level = 11. (c) Noise level = 17. (d) Polygonal approximation for (a). (e) Polygonal approximation for (b). (f) Polygonal approximation for (c).

many of the symbolic features. When feature values obtained for a specific feature (e.g., wing span or leading wing angle) vary significantly on opposite sides of an aircraft (usually due to distortion in the aircraft image), the feature is not extracted due to the ambiguity of the situation. This approach insures that target misclassification does not result from the presence of uncertain feature information. Once all possible features have been extracted from the polygonal approximation, the symbolic feature set is ready for processing by the TRIPLE system.

In the examples described below, the image distortion, border following, and polygonal approximation algorithms have all been implemented and executed on a SUN 3/60 workstation. The polygonal approximation data is transferred to a Symbolics 3670 workstation, which performs the symbolic feature extraction operation and hosts the TRIPLE target recognition system.

3.1 System Training

The first step in the target recognition process is to construct an initial collection of target models. As described in Section 2, this operation is automated in the TRIPLE system. The user merely supplies a set of training images, which are processed by the TRIPLE system to generate a set of target models. Figure 4 shows the initial set of aircraft used to initialize the TRIPLE system. These aircraft have not been distorted since most training operations utilize high quality training data to insure accuracy. Figure 5 provides the set of

symbolic target features obtained for each of the aircraft in Figure 4. Missing features are due to inconsistencies in feature values or other aircraft anomalies.

Once the symbolic feature information for all training examples is available, the TRIPLE system must construct the target classification tree. To do so, the EBL component is invoked on each of the aircraft models to select the set of relevant target model features. EBL sequentially processes each of the symbolic features lists shown in Figure 5 using the information in the background knowledge base. The knowledge base for these examples consists of a generic aircraft prototype that specifies the presence of wing, fuselage, engine, and tail features in order to generate an aircraft target model. The knowledge base contains 23 different rules that are utilized to establish allowable combinations of the features to satisfy the wing, fuselage, engine, and tail requirements. Figure 6 illustrates the resulting target models created by the EBL component for the system initialization phase. Notice that the specific feature values have been generalized into ranges of values and that EBL has generated a weight associated with each feature in the target model. The weights are used during matching to compute target recognition confidence.

Following the selection of relevant target features by the EBL component, the TRIPLE system invokes the SCC process to construct the initial target classification tree. All seven target models are given to SCC, which builds the classification tree shown in Figure 7. The nodes in the tree

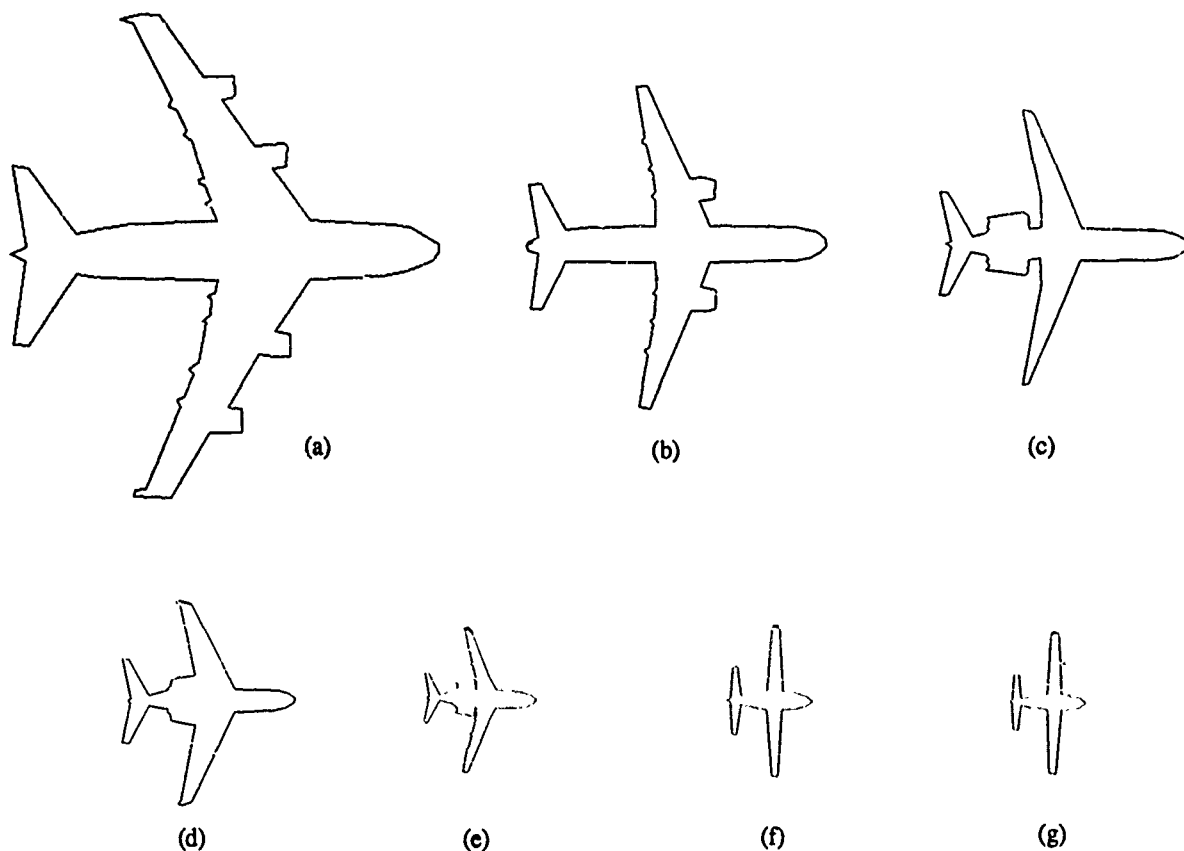


Figure 4: Aircraft used during the initialization phase of the TRIPLE system. (a) Boeing 747 (B-747). (b) Boeing 757 (B-757). (c) McDonnell Douglas MD-87 (MD-87). (d) Gulfstream Aerospace (Aerospace). (e) Cessna Citation (Citation). (f) Cessna Caravan (Caravan). (g) Piper Malibu (Malibu).

Feature	B-747	B-757	MD-87	Aerospace	Citation	Caravan	Malibu
Wingspan	243'	160'	140'	103'	69'	67'	53'
Wing Sweep, Leading	129°	114°	115°	119°	116°	95°	97°
Wing Sweep, Trailing	112°	97°	99°	102°	101°	88°	86°
Wing Base Chord	53'	28'	22'	22'	10'	7'	6'
Wing Tip Chord	12'	6'	4'	---	3'	---	3'
Wing Taper, Base/Tip	4.37	4.86	6.00	---	3.33	---	2.26
Fuselage Length	222'	155'	130'	88'	55'	38'	32'
Fuselage Width	27'	18'	16'	11'	8'	8'	7'
Length, Wing-to-Nose	62'	60'	58'	32'	20'	12'	11'
Length, Wing-to-Tail	75'	47'	---	---	---	14'	10'
Nose Shape	ROUND	---	ROUND	ROUND	---	---	POINTED
Position of Engines	ON-WING	ON-WING	FUSELAGE	FUSELAGE	FUSELAGE	NOSE	NOSE
Number of Engines	4	2	2	2	2	1	1
Tailspan	89'	62'	52'	42'	24'	26'	24'
Tail Sweep, Leading	126°	118°	120°	120°	120°	96°	97°
Tail Sweep, Trailing	99°	99°	102°	102°	101°	87°	87°
Tail Base Chord	28'	15'	12'	10'	6'	5'	4'
Tail Tip Chord	8'	6'	4'	4'	2'	3'	2'
Tail Taper, Base/Tip	4.03	2.40	2.92	2.73	3.17	1.88	2.40
Wingspan/Tailspan	2.76	2.59	2.68	2.45	2.91	2.53	2.22

Figure 5: Symbolic features extracted from the aircraft in Figure 4.

Feature	Value	Weight
Wingspan	(Range 239'-251')	0.10
Wing Sweep, Leading	(Range 127°-131°)	0.08
Wing Sweep, Trailing	(Range 110°-114°)	0.07
Wing Base Chord	(Range 50'-56')	0.06
Wing Tip Chord	(Range 11'-13')	0.04
Fuselage Length	(Range 216'-228')	0.12
Fuselage Width	(Range 25'-29')	0.04
Length, Wing-to-Nose	(Range 59'-65')	0.06
Length, Wing-to-Tail	(Range 72'-78')	0.08
Position of Engines	ON-WING	0.10
Number of Engines	4	0.05
Tailspan	(Range 86'-92')	0.08
Tail Sweep, Leading	(Range 124°-128°)	0.06
Tail Sweep, Trailing	(Range 97°-101°)	0.04
Tail Base Chord	(Range 26'-30')	0.02

(a)

Feature	Value	Weight
Wingspan	(Range 155'-165')	0.11
Wing Sweep, Leading	(Range 112°-116°)	0.09
Wing Sweep, Trailing	(Range 95°-99°)	0.09
Wing Base Chord	(Range 26'-30')	0.06
Fuselage Length	(Range 150'-160')	0.12
Fuselage Width	(Range 17'-19')	0.04
Length, Wing-to-Nose	(Range 57'-63')	0.06
Length, Wing-to-Tail	(Range 45'-49')	0.08
Position of Engines	ON-WING	0.10
Number of Engines	2	0.05
Tailspan	(Range 59'-65')	0.08
Tail Sweep, Leading	(Range 116°-120°)	0.06
Tail Sweep, Trailing	(Range 97°-101°)	0.04
Tail Base Chord	(Range 14'-16')	0.02

(b)

Feature	Value	Weight
Wingspan	(Range 136'-144')	0.11
Wing Sweep, Leading	(Range 113°-117°)	0.09
Wing Sweep, Trailing	(Range 97'-101°)	0.09
Wing Base Chord	(Range 21'-23')	0.06
Fuselage Length	(Range 126'-134')	0.14
Fuselage Width	(Range 15'-17')	0.06
Length, Wing-to-Nose	(Range 55'-61')	0.10
Position of Engines	FUSELAGE	0.10
Number of Engines	2	0.05
Tailspan	(Range 49'-55')	0.08
Tail Sweep, Leading	(Range 118°-122°)	0.06
Tail Sweep, Trailing	(Range 100°-104°)	0.04
Tail Base Chord	(Range 11'-13')	0.02

(c)

Feature	Value	Weight
Wingspan	(Range 99'-107')	0.11
Wing Sweep, Leading	(Range 117°-121°)	0.09
Wing Sweep, Trailing	(Range 100°-104°)	0.09
Wing Base Chord	(Range 21'-23')	0.06
Fuselage Length	(Range 85'-91')	0.14
Fuselage Width	(Range 10'-12')	0.06
Length, Wing-to-Nose	(Range 30'-34')	0.10
Position of Engines	FUSELAGE	0.10
Number of Engines	2	0.05
Tailspan	(Range 40'-44')	0.08
Tail Sweep, Leading	(Range 118°-122°)	0.06
Tail Sweep, Trailing	(Range 100°-104°)	0.04
Tail Base Chord	(Range 9'-11')	0.02

(d)

Feature	Value	Weight
Wingspan	(Range 66'-72')	0.11
Wing Sweep, Leading	(Range 114°-118°)	0.09
Wing Sweep, Trailing	(Range 99°-103°)	0.09
Wing Base Chord	(Range 9'-11')	0.06
Fuselage Length	(Range 52'-58')	0.18
Length, Wing-to-Nose	(Range 19'-21')	0.12
Position of Engines	FUSELAGE	0.10
Number of Engines	2	0.05
Tailspan	(Range 23'-25')	0.08
Tail Sweep, Leading	(Range 118°-122°)	0.06
Tail Sweep, Trailing	(Range 99°-103°)	0.06

(e)

Feature	Value	Weight
Wingspan	(Range 64'-70')	0.17
Wing Sweep, Leading	(Range 93°-97°)	0.10
Wing Sweep, Trailing	(Range 86°-90°)	0.08
Fuselage Length	(Range 36'-40')	0.12
Length, Wing-to-Nose	(Range 11'-13')	0.08
Length, Wing-to-Tail	(Range 13'-15')	0.10
Position of Engines	NOSE	0.10
Number of Engines	1	0.05
Tailspan	(Range 24'-28')	0.08
Tail Sweep, Leading	(Range 94°-98°)	0.06
Tail Sweep, Trailing	(Range 85°-89°)	0.06

(f)

Feature	Value	Weight
Wingspan	(Range 50'-56')	0.17
Wing Sweep, Leading	(Range 95°-99°)	0.10
Wing Sweep, Trailing	(Range 84°-88°)	0.08
Fuselage Length	(Range 30'-34')	0.12
Length, Wing-to-Nose	(Range 10'-12')	0.08
Length, Wing-to-Tail	(Range 9'-11')	0.10
Position of Engines	NOSE	0.10
Number of Engines	1	0.05
Tailspan	(Range 23'-25')	0.08
Tail Sweep, Leading	(Range 95°-99°)	0.06
Tail Sweep, Trailing	(Range 85°-89°)	0.06

(g)

Figure 6: EBL-generated target models for each of the symbolic feature lists shown in Figure 5. (a) B-747. (b) B-757. (c) MD-87. (d) Aerospace. (e) Citation. (f) Caravan. (g) Malibu.

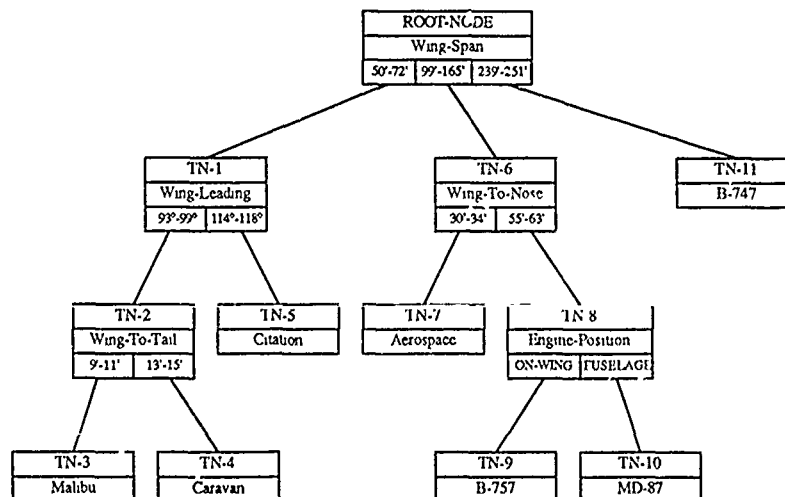


Figure 7: SCC-generated target classification tree.

are labeled *TN**, which stands for *TREE-NODE**. Note that the aircraft have been effectively segmented into intuitively obvious groups by the SCC component.

3.2 Complete Recognition

The classification tree shown in Figure 7 is used by the TRIPLE system to recognize subsequent instances of the aircraft which have been modeled during training. Figure 8 shows an example of an "unknown" aircraft (*Malibu*) that must be recognized by the target recognition system. The aircraft image (Figure 8(a)) is moderately distorted and thus, in the polygonal approximation (Figure 8(b)), it is more irregular than the training example. The distortion of the aircraft becomes apparent by analyzing the list of extracted symbolic features, shown in Figure 8(c), and comparing this list with the previous collection of features shown in Figure 6(g). Only a few of the tail features are available due to the aircraft distortion.

The model matching component uses the list of features in Figure 8(c) to parse the classification tree in Figure 7. At the *ROOT-NODE*, the unknown aircraft is compatible with the leftmost branch, so the matching component traverses the tree to node *TN-1*. At this location, the unknown object matches the leftmost branch again, so the matching process moves to node *TN-2*. Here, the matching process must consider two possible alternatives due to the fact that the wing-to-tail feature is missing from the unknown aircraft. Both branches are investigated by the matching process to determine whether either of them (or possibly both) are compatible with the unknown target. The right branch of *TN-2* is discounted due to differences in wing span, fuselage length, wing-to-nose, tail span, and tail leading angle. However, the left branch, *TN-3*, which contains the *Malibu* aircraft model, is found to be compatible with the unknown aircraft. The matching confidence of this target model is computed to be 74.6%. The confidence is derived using the weights assigned to each target model feature and the error between the feature values in the target model and the unknown aircraft. Even though the aircraft feature set was missing two features specified in the *Malibu* target model (wing-to-tail and tail leading angle), the TRIPLE system was able to correctly recognize the aircraft.

Additionally, since the recognition confidence of the aircraft is greater than the complete recognition threshold (70% for these experiments), the feature value monitor is invoked to update the values in the *Malibu* aircraft model. The revised *Malibu* model is shown in Figure 8(d).

3.3 Incomplete Recognition

The TRIPLE system is capable of partially identifying an unknown aircraft when very few symbolic features are available to describe the target. Figure 9 provides an example of an aircraft that causes the system to produce an incomplete recognition result. In this image (Figure 9(a)), the tail of the aircraft and the engine regions have been separated from the main portion of the target. This situation commonly occurs in cases where there is low contrast between the aircraft and the background or in cases where shadows or surfaces markings on the target blend in with the background. Since the border following algorithm is designed to locate only the largest target region, the system creates the polygonal approximation shown in Figure 9(b). The set of symbolic aircraft features which can be extracted from this result are indicated in Figure 9(c). Due to the lack of any tail information and discrepancies in the wing representation, very few reliable features have been obtained for this target.



(a)



(b)

Feature	Value
Wingspan	54'
Wing Sweep, Leading	97°
Wing Sweep, Trailing	87°
Wing Base Chord	5'
Wing Tip Chord	3'
Wing Taper, Base/Tip	1.85
Fuselage Length	30'
Fuselage Width	7'
Length, Wing-to-Nose	10'
Position of Engines	NOSE
Number of Engines	1
Tailspan	23'
Tail Sweep, Trailing	87°
Tail Tip Chord	2'
Wingspan/Tailspan	2.35

(c)

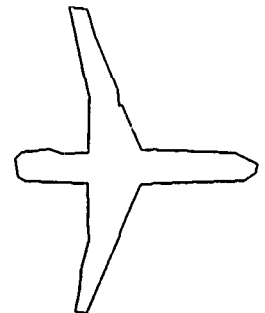
Feature	Value	Weight
Wingspan	(Range 51'-57')	0.17
Wing Sweep, Leading	(Range 95°-99°)	0.10
Wing Sweep, Trailing	(Range 85°-89°)	0.08
Fuselage Length	(Range 29'-33')	0.12
Length, Wing-to-Nose	(Range 9'-11')	0.08
Length, Wing-to-Tail	(Range 9'-11')	0.10
Position of Engines	NOSE	0.10
Number of Engines	1	0.05
Tailspan	(Range 22'-24')	0.08
Tail Sweep, Leading	(Range 95°-99°)	0.06
Tail Sweep, Trailing	(Range 85°-89°)	0.06

(d)

Figure 8: Aircraft (*Malibu*) which illustrates the complete recognition state of the TRIPLE system. (a) Distorted aircraft image. (b) Polygonal approximation of the aircraft. (c) Symbolic target features extracted from the aircraft. (d) Revised *Malibu* aircraft model after complete recognition cycle.



(a)



(b)

Feature	Value
Wing Sweep, Leading	114°
Wing Sweep, Trailing	97°
Wing Tip Chord	4'
Fuselage Width	17'
Length, Wing-to-Nose	60'
Nose Shape	ROUND

(c)

Figure 9: Aircraft (*B-757*) which illustrates the incomplete recognition state of the TRIPLE system. (a) Distorted aircraft image. (b) Polygonal approximation of the aircraft. (c) Symbolic target features extracted from the aircraft.

The matching component begins at the *ROOT-NODE* of the classification tree (Figure 7) by inspecting the wing span value of the unknown aircraft. Because the wing span is missing, the matching component hypothesizes all three branches of the tree (Nodes *TN-1*, *TN-6*, and *TN-11*) as possible alternatives. *TN-11*, which contains the *B-747* aircraft model, is rejected due to differences in every single target model feature except wing-to-nose. At *TN-1*, the unknown aircraft's leading wing angle is compatible with the right branch of the node, so parsing continues down to *TN-5*. However, the *Citation* aircraft model contained in *TN-5* conflicts with the unknown aircraft in every feature except the leading wing angle. Thus, this hypothesis is also rejected.

Looking at *TN-6*, the matching process selects the right branch and moves to the *TN-8* tree node. Since the engine position feature is missing, the matching process once again considers both branches as possible alternatives. Inspecting *TN-9*, the matching process finds that the *B-757* aircraft model is compatible with the unknown target (matching confidence = 25.9%). At *TN-10*, the unknown aircraft is also correctly matched to the *MD-87* aircraft model (matching confidence = 27.5%). Since no additional feature information is available to select between these two alternatives, the *TRIPLE* system reports both aircraft models as possible matches.

3.4 Occluded Recognition

Target occlusion can be effectively handled by the model matching process performed in the *TRIPLE* system. Occluded recognition performance in *TRIPLE* system is very similar in nature to incomplete recognition. The difference between the two cases is that the missing target features tend to be the global features in the case of occlusion whereas, in the case of incomplete recognition, the missing features are usually the local target features.

Figure 10(a) provides an example of an aircraft image that illustrates the occluded recognition scenario. In this example, the nose and the port wing of the aircraft, which is an instance of the *MD-87* target model, have been occluded. The polygonal approximation of this target is shown in Figure 10(b). The symbolic feature extraction process is still able to derive a useful set of features from the aircraft, as indicated in Figure 10(c).

The model matching component uses the list of symbolic feature information to parse the classification tree shown in Figure 7. At the *ROOT-NODE*, the wing span value of the unknown aircraft is compatible with the center branch, so the matching component proceeds down to node *TN-6*. The wing-to-nose feature is missing in the feature list, so both branches (*TN-7* and *TN-8*) are hypothesized. Examining *TN-7*, the model matching process finds that the wing span and tail span feature values contradict those of the *Aerospace* target model stored in the node, although all other features are compatible. Thus, *TN-7* is discarded. At *TN-8*, the right branch of the node is compatible with the engine position feature in the feature list. Finally, at node *TN-10*, the matching process discovers that the *MD-87* target model is compatible with the feature list. The recognition confidence in this example is 65.9%. No changes are made to the target model since the recognition confidence is below the complete recognition confidence threshold.

3.5 Target Model Acquisition

The machine learning capabilities of the *TRIPLE* system are evident in the target model acquisition and model refinement operations performed by the system. This section

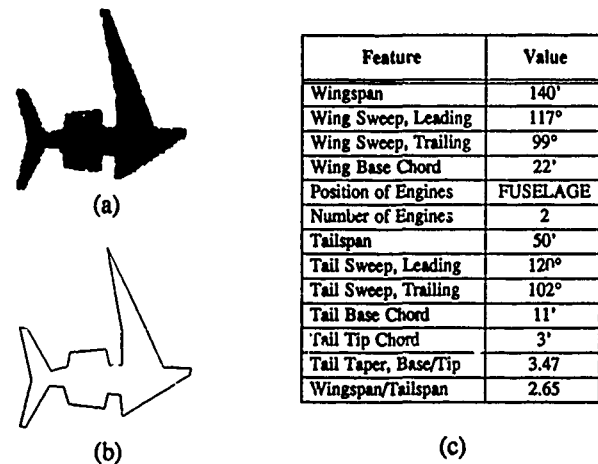


Figure 10: Aircraft (MD-87) which illustrates the occluded recognition state of the *TRIPLE* system. (a) Distorted aircraft image. (b) Polygonal approximation of the aircraft. (c) Symbolic target features extracted from the aircraft.

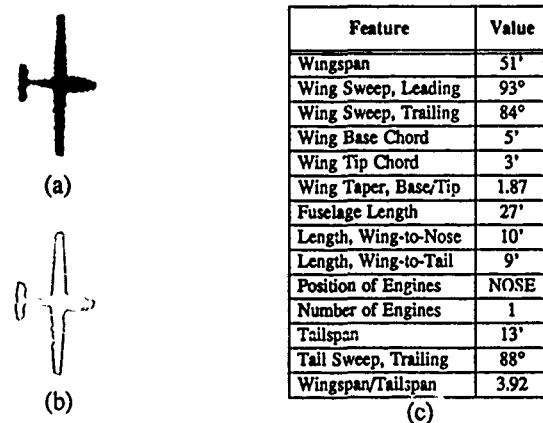


Figure 11: Aircraft (Renegade) which illustrates the target model acquisition capabilities of the *TRIPLE* system. (a) Distorted aircraft image. (b) Polygonal approximation of the aircraft. (c) Symbolic target features extracted from the aircraft. (d) EBL-generated target model for the aircraft.

Feature	Value	Weight
Wingspan	(Range 48'-54')	0.17
Wing Sweep, Leading	(Range 91°-95°)	0.10
Wing Sweep, Trailing	(Range 82°-86°)	0.08
Fuselage Length	(Range 25'-29')	0.12
Length, Wing-to-Nose	(Range 9'-11')	0.08
Length, Wing-to-Tail	(Range 8'-10')	0.10
Position of Engines	NOSE	0.10
Number of Engines	1	0.05
Tailspan	(Range 12'-14')	0.20

(d)

demonstrates several examples of the automated target model acquisition scenario.

Figure 11(a) shows an image of an unknown aircraft. In this case, the aircraft is a Lake Renegade (*Renegade*) which has never been seen by the target recognition system. Figure 11(b) illustrates the polygonal approximation of the aircraft image and Figure 11(c) provides the list of symbolic target features extracted from the polygonal representation. As with any other unknown object, the *TRIPLE* system begins by

parsing the classification tree (Figure 7) using the set of symbolic target features. Traversing the tree in standard fashion, the knowledge-based matching component arrives at node TN-3 and compares the symbolic feature list with the *Malibu* target model. However, differences in leading wing angle, trailing wing angle, fuselage length, and tail span cause the *Malibu* target model to be discarded. Since no other branches in the tree were hypothesized during parsing, the current classification tree contains insufficient information to identify this aircraft.

The EBL process is subsequently invoked in an attempt to acquire the unknown aircraft as a new target model. Figure 11(d) illustrates the EBL-generated target model produced from the symbolic feature list in Figure 11(c). The new target model is then handed to the SCC component so that it can be incorporated in the classification tree structure. SCC parses the tree using the new target model in an attempt to leave as much of the tree intact as possible. SCC successfully traverses the tree until it encounters the TN-3 leaf node. The tree is reclustered at this point to distinguish between the current *Malibu* target model and the new *Renegade* target model. Tail span is found to be the best symbolic feature that separates the two target models. The revised classification tree, after insertion of the *Renegade* target model, is shown in Figure 12.

Figure 13 presents a second, more complex example of the target model acquisition process. Figures 13(a) and 13(b) show an image of an unknown aircraft and the corresponding polygonal approximation of this image. The aircraft in this case is a McDonnell Douglas MD-11 (*MD-11*). The extracted list of symbolic target features is shown in Figure 13(c). As before, the process begins by parsing the current classification tree (Figure 12). The parsing process immediately terminates since the wing span value of the unknown target is not compatible with any of the branches at the *ROOT-NODE*.

EBL is called upon to acquire the aircraft as a new target model. In this case, the acquisition process succeeds, as indicated by the new target model shown in Figure 13(d).

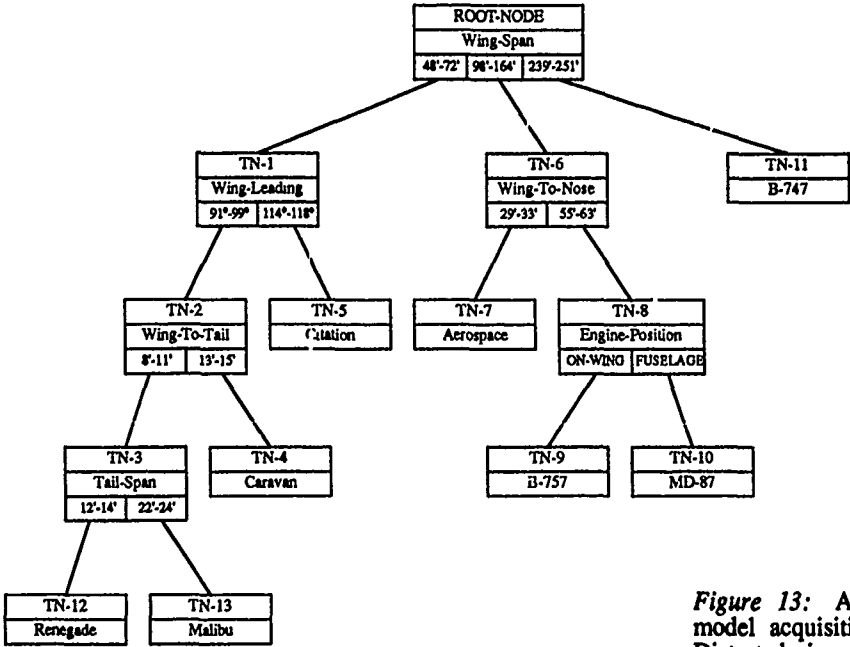
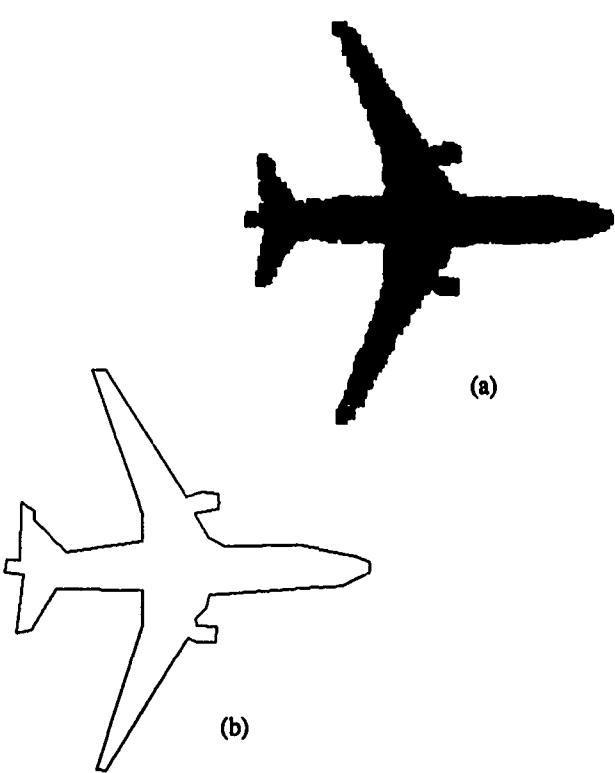


Figure 12: Revised target classification tree, after insertion of the Renegade aircraft model.



Feature	Value
Wingspan	208'
Wing Sweep, Leading	127°
Wing Sweep, Trailing	106°
Wing Base Chord	37'
Fuselage Length	199'
Fuselage Width	27'
Length, Wing-to-Nose	84'
Nose Shape	ROUND
Position of Engines	ON-WING
Number of Engines	2
Tailspan	68'
Tail Sweep, Trailing	100°
Tail Base Chord	20'
Tail Tip Chord	8'
Tail Taper, Base/Tip	2.17
Wingspan/Tailspan	3.01

(c)

Feature	Value	Weight
Wingspan	(Range 202'-214')	0.11
Wing Sweep, Leading	(Range 125°-129°)	0.09
Wing Sweep, Trailing	(Range 104°-108°)	0.09
Wing Base Chord	(Range 35'-39')	0.06
Fuselage Length	(Range 194'-204')	0.14
Fuselage Width	(Range 25'-29')	0.06
Length, Wing-to-Nose	(Range 81'-87')	0.10
Position of Engines	ON-WING	0.10
Number of Engines	2	0.05
Tailspan	(Range 65'-71')	0.20

(d)

Figure 13: Aircraft (MD-11) which illustrates the target model acquisition capabilities of the TRIPLE system. (a) Distorted aircraft image. (b) Polygonal approximation of the aircraft. (c) Symbolic target features extracted from the aircraft. (d) EBL-generated target model for the aircraft.

This model is then passed to SCC so that it can be added to the classification tree. Since none of the branches at the *ROOT-NODE* are compatible with the wing span of the new model, SCC is forced to recluster the tree at the root level. In doing so, the SCC process discovers that fuselage length is now a better distinguishing feature than wing span at the root node in the tree. The resulting target classification tree, after reclustering has been completed, is shown in Figure 14. The new *MD-11* target model has been included in the same branch as the *B-747*, with wing-to-nose used as the distinguishing feature.

3.6 Target Model Refinement

The second machine learning capability of the TRIPLE system is present in the automated target model refinement process described in this section. Target model refinement occurs when the presence of a new symbolic feature is detected in an aircraft that can be correctly recognized by the target recognition system. The EBL-SCC learning cycle is invoked in these instances to determine if the new feature is relevant in recognizing the aircraft and, if so, where it should be placed in the target classification tree. Several examples of target model refinement are now presented. The classification tree used in these experiments is shown in Figure 15. This tree was obtained from the tree in Figure 14 after two more aircraft models (*Metro* and *Learjet*) were acquired.

The first aircraft to be refined by the TRIPLE system is shown in Figure 16(a). This aircraft is an instance of the *MD-11* target model that was acquired in Section 3.5. The polygonal approximation and the list of symbolic target features are shown in Figures 16(b) and 16(c), respectively. The model matching process uses the symbolic features to parse the classification tree in Figure 15. The unknown aircraft is correctly identified as an *MD-11* aircraft with a recognition confidence of 94.3%. In addition, the model matching process detects the presence of two new features in the unknown aircraft, wing-to-tail and leading tail angle. Both of these features were missing from the feature list in Figure 13 due to errors in the polygonal approximation of the aircraft

caused by image distortion.

Since new features are present in the correctly identified aircraft, the EBL-SCC learning cycle is entered to ascertain the relevance of the features. The two new features are added to the current *MD-11* target model and the entire list of features is reprocessed by the EBL component. In this example, EBL does create a new target model (Figure 16(d)) since wing-to-tail and leading tail angle were found to be significant. Further, the EBL process has selected trailing tail angle and tail-base as additional relevant features in conjunction with the presence of the leading tail angle feature.

The revised *MD-11* target model is sent to the SCC component in order to update the classification tree. At the *ROOT-NODE*, the four new target features are compared with the fuselage length feature to see if they produce a better conceptual clustering of the target models at that position in the tree. None of them do, so the process repeats at *TN-26* by comparing the clustering quality of the wing-to-nose feature with the new relevant features. Once again, the tree node is left intact and finally, the new target features are simply added to the list of relevant features at the *TN-28* leaf node.

Figure 17 provides another example of the model refinement process. The image of a *Renegade* aircraft is shown in Figure 17(a) and the corresponding polygonal approximation is indicated in Figure 17(b). The list of symbolic target features obtained from this aircraft are presented in Figure 17(c). The tree is parsed using this feature information and, although the matching process must hypothesize nodes *TN-36* and *TN-39* during the tree traversal, the aircraft is finally identified as an instance of the *Renegade* target model. The recognition confidence in this case is 86.1%. The nose shape, leading tail angle, and tail base features in Figure 17(c) are discovered to be new model features and thus, the target model refinement operation is invoked.

EBL processes the *Renegade* aircraft model using the three new target features. In this case, the leading tail angle is found to be relevant along with the trailing tail angle that was present, but not relevant, in the initial *Renegade* sym-

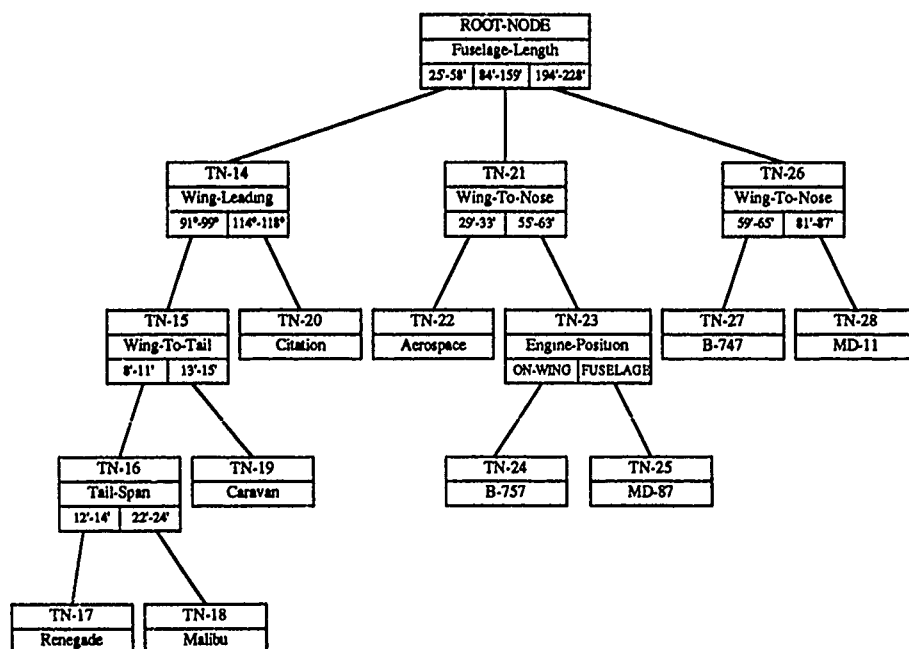


Figure 14: Revised target classification tree, after insertion of the MD-11 aircraft model.

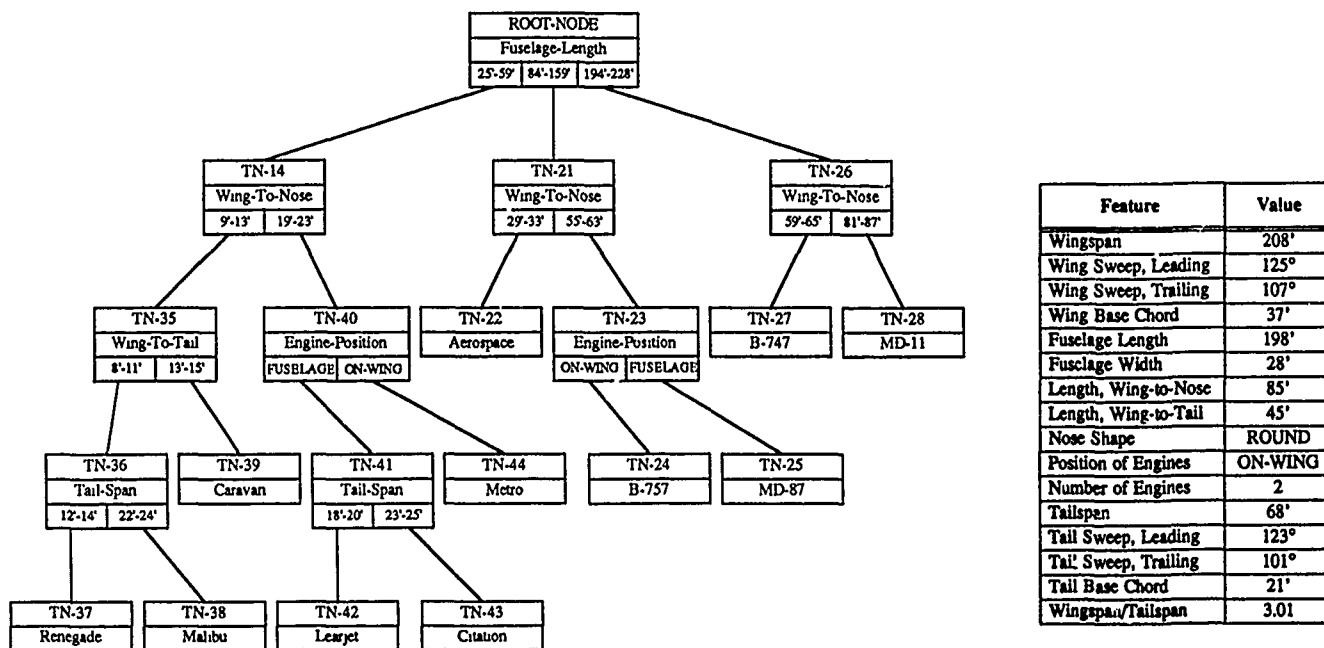
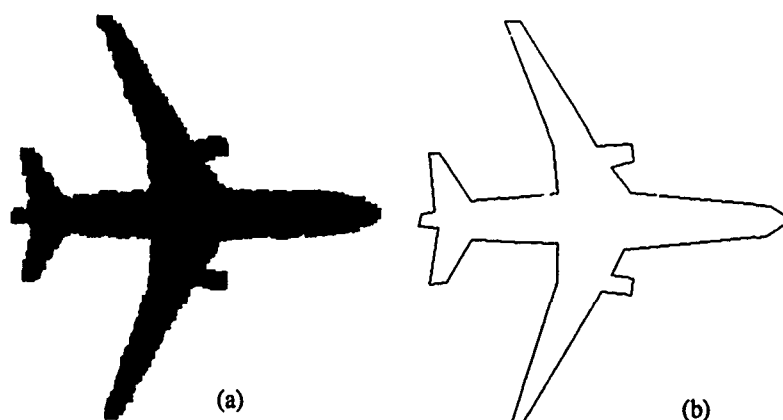


Figure 15: Target classification tree for the target model refinement experiments.



Feature	Value	Weight
Wingspan	(Range 202'-214')	0.11
Wing Sweep, Leading	(Range 124°-128°)	0.09
Wing Sweep, Trailing	(Range 105°-109°)	0.09
Wing Base Chord	(Range 35'-39')	0.06
Fuselage Length	(Range 193'-203')	0.12
Fuselage Width	(Range 26'-30')	0.04
Length, Wing-to-Nose	(Range 82'-88')	0.06
Length, Wing-to-Tail	(Range 43'-47')	0.08
Position of Engines	ON-WING	0.10
Number of Engines	2	0.05
Tailspan	(Range 65'-71')	0.08
Tail Sweep, Leading	(Range 121°-125°)	0.06
Tail Sweep, Trailing	(Range 98°-102°)	0.04
Tail Base Chord	(Range 19'-21')	0.02

(d)

Figure 16: Aircraft (MD-11) which illustrates the target model refinement capabilities of the TRIPLE system. (a) Distorted aircraft image. (b) Polygonal approximation of the aircraft. (c) Symbolic target features extracted from the aircraft. (d) EBL-generated target model for the aircraft.



(c)

Feature	Value
Wingspan	50'
Wing Sweep, Leading	93°
Wing Sweep, Trailing	85°
Wing Base Chord	5'
Wing Tip Chord	3'
Wing Taper, Base/Tip	2.08
Fuselage Length	28'
Length, Wing-to-Nose	10'
Nose Shape	ROUND
Position of Engines	NOSE
Number of Engines	1
Tailspan	13'
Tail Sweep, Leading	89°
Tail Sweep, Trailing	89°
Tail Base Chord	4'
Wingspan/Tailspan	3.68

Feature	Value	Weight
Wingspan	(Range 48'-52')	0.17
Wing Sweep, Leading	(Range 91°-95°)	0.10
Wing Sweep, Trailing	(Range 83°-87°)	0.08
Fuselage Length	(Range 26'-30')	0.12
Length, Wing-to-Nose	(Range 9'-11')	0.08
Length, Wing-to-Tail	(Range 8'-10')	0.10
Position of Engines	NOSE	0.10
Number of Engines	1	0.05
Tailspan	(Range 12'-14')	0.08
Tail Sweep, Leading	(Range 87°-91°)	0.06
Tail Sweep, Trailing	(Range 86°-90°)	0.06

(d)

Figure 17: Aircraft (Renegade) which illustrates the target model refinement capabilities of the TRIPLE system. (a) Distorted aircraft image. (b) Polygonal approximation of the aircraft. (c) Symbolic target features extracted from the aircraft. (d) EBL-generated target model for the aircraft.

bolic feature list. The revised target model is shown in Figure 17(d). As in the previous model refinement example, SCC is given the revised model for insertion into the classification tree. SCC finds that at *TN-36*, the new leading tail angle is a better distinguishing feature than the current tail span feature (Figure 15), so the classification tree is reclustered at *TN-36*. The final structure of the classification tree, after the target model refinement process, is shown in Figure 18.

3.7 Recognition Failure

This section presents a brief example of recognition failure in the TRIPLE system. As with any target recognition system, there will always be instances where the information processed by the system or the knowledge used to process the information is insufficient to perform the recognition task. This example demonstrates how incomplete feature information leads to recognition failure in the TRIPLE system.

Figure 19(a) provides an aircraft image (a Fairchild Merlin aircraft which has not previously been modeled) that must be identified by the recognition system. The polygonal approximation of the aircraft (Figure 19(b)) contains only the front part of the aircraft due to the separation in the fuselage portion of the image. The symbolic feature list obtained from this approximation is shown in Figure 19(c). The knowledge-based matching component uses the feature data to parse the classification tree in Figure 18. At the *ROOT-NODE*, the wing-span value is missing, so nodes *TN-14*, *TN-21*, and *TN-26* are hypothesized. However, at each of these nodes, none of the available branches are compatible with the aircraft's feature data, so the model matching process terminates. EBL is invoked to acquire the new aircraft, but is unable to generate an acceptable target model due to the absence of any tail features. Since EBL can not process the available feature data, the aircraft is reported as a recognition failure.

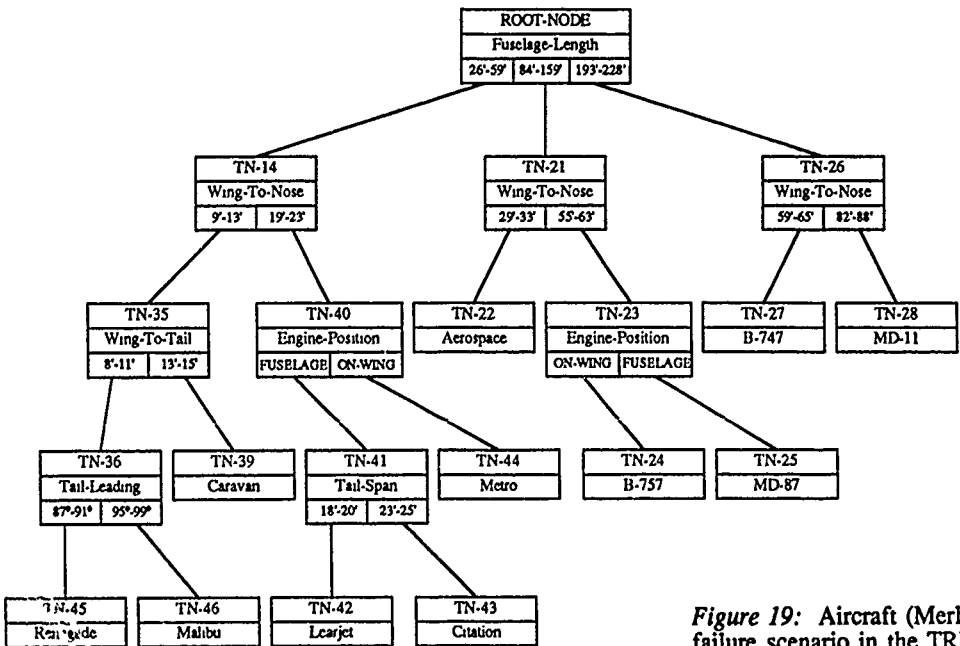


Figure 18: Revised target classification tree, after refinement of the Renegade aircraft model.

4. CURRENT RESEARCH

The experiments performed using the TRIPLE system to date have served as a proof of concept for the integrated target recognition/machine learning approach. We are now involved in extending the capabilities of the TRIPLE system to handle complex 3D object descriptions. These changes involve modifications to the segmentation and symbolic feature extraction component to obtain valid symbolic features for 3D objects seen at arbitrary angles.

Figure 20(a) presents a typical example of an image containing an aircraft and Figure 20(b) indicates a segmented view of this image in which the aircraft regions are prominent. Due to the oblique angle imagery, the aircraft can not be matched using a 2D target model. Instead, the symbolic feature extraction component now utilizes a generic aircraft description similar to the one shown in Figure 21. By hypothesizing various orientations of the 3D aircraft prototype and predicting the appearance of specific target features, the necessary target features can be derived from the segmentation results shown in Figure 20(b).

Once the 3D symbolic features have been obtained using this process, the TRIPLE system can process the data and recognize the unknown aircraft in a similar fashion to the matching approach described in this paper. By extending the background knowledge base to handle the additional 3D target features, the TRIPLE system can recognize, acquire, and refine complex, 3D target models. Work is currently underway to refine and implement these concepts.

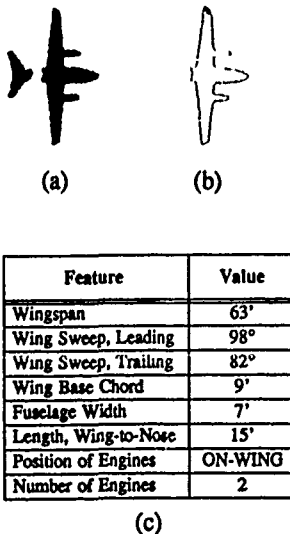


Figure 19: Aircraft (Merlin) which illustrates the recognition failure scenario in the TRIPLE system. (a) Distorted aircraft image. (b) Polygonal approximation of the aircraft. (c) Symbolic target features extracted from the aircraft.



(a)



(b)

Figure 20: Typical imagery for the 3D target recognition experiments. (a) Oblique angle image of two aircraft. (b) Segmented aircraft regions used during symbolic feature extraction.

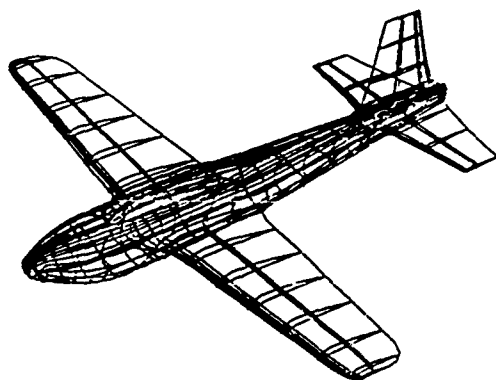


Figure 21: Generic aircraft prototype used to obtain symbolic object features in the 3D target recognition experiments.

5. CONCLUSIONS

We have presented the experimental results of a new target recognition system that exhibits the standard target recognition system functionalities (complete recognition, partial recognition, occluded recognition) as well as providing several new capabilities (target model acquisition and target model refinement). The machine learning components built into the TRIPLE system allow it to adapt its representation of the individual target models in order to operate effectively in an unconstrained, dynamic environment. The TRIPLE system is part of a complete multilevel machine learning system for target recognition that we are developing.¹

REFERENCES

1. B. Bhanu, "Image Understanding Research at Honeywell," Proceedings of DARPA Image Understanding Workshop (September, 1990).
2. B. Bhanu and J.C. Ming, "TRIPLE: A Multi-Strategy Machine Learning Approach to Target Recognition," Proceedings of DARPA Image Understanding Workshop, pp. 537-547 (April, 1988).
3. G. DeJong and R. Mooney, "Explanation-Based Learning: An Alternative View," *Machine Learning* 1(2) pp. 145-176 (1986).
4. R.S. Michalski, "Knowledge Acquisition through Conceptual Clustering: A Theoretical Framework and Algorithm for Partitioning Data into Conjunctive Concepts," *International Journal of Policy Analysis and Information Systems* 4 pp. 219-243 (1980).
5. R.S. Michalski and R.E. Stepp, "Automated Construction of Classifications: Conceptual Clustering Versus Numerical Taxonomy," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 5(4) pp. 396-409 (1983).
6. T.M. Mitchell, R.M. Keller, and S.T. Kedar-Cabelli, "Explanation-Based Generalization: A Unifying View," *Machine Learning* 1(1) pp. 47-80 (1986).
7. R.E. Stepp and R.S. Michalski, "Conceptual Clustering of Structured Objects: A Goal-Oriented Approach," *Artificial Intelligence* 28(1) pp. 43-69 (1986).

A Formalization and Implementation of Topological Visual Navigation in Two Dimensions

John R. Kender

Il-Pyung Park

David Yang

Department of Computer Science
Columbia University
New York, New York 10027

Abstract

In this paper we formalize and implement a model of topological visual navigation in two-dimensional spaces. Unlike much of traditional quantitative visual navigation, the emphasis throughout is on the methods and the efficiency of qualitative visual descriptions of objects and environments, and on the methods and the efficiency of direction-giving by means of visual landmarks. We formalize three domains—the world itself, the map-maker's view of it, and the navigator's experience of it—and the concepts of custom maps and landmarks. We specify, for a simplified navigator (the "level helicopter") the several ways in which visual landmarks can be chosen, depending on which of several costs (sensor, distance, or communication) should be minimized. We show that paths minimizing one measure can make others arbitrarily complex; the algorithm for selecting the path is based on a form of Dijkstra's algorithm, and therefore automatically generates intelligent navigator overshooting and backtracking. We implement, using an arm-held camera, such a navigator, and detail its basic seek-and-adjust behaviors as it follows visual highways (or departs from them) to reach a goal. Seeking is based on topology, and adjusting is based on symmetry; there are essentially no quantitative measures. We describe under what circumstances its environment is visually difficult and perceptively shadowed, and describe how errors in path-following impact landmark selection. Since visual landmark selection and direction-giving are in general NP-complete, and rely on the nearly intractable concept of characteristic views, we suggest some heuristics; one is that the landmark object "itself", rather than its views, may be its most compact encoding. We conclude with speculations about the feasibility of intelligent navigation in very large self-occluding visual worlds.

1 Introduction

In this paper, we define a model and describe an implementation for characterizing a class of visual navigation problems in-the-large. Although both model and implementation are extendible, this paper focuses on visual landmark selection and topological navigation within (actually, above) a two-dimensional environment, without the use of metric distance information, and without the use of "graphic" information such as street signs. The domain we explore is a formalized variant of the visual world seen by a level-flying helicopter whose perception is limited to a narrow field of view, and whose visual axis is normal to the world below it. Thus, occlusions are not yet an issue, but object identity and confusability are, and so are issues of communicative, perceptive, and motive economy.

This work is ultimately motivated by the questions of what is a good map, what is a good landmark, and what is a good visual environment. It addresses the abstract problem of how to optimally choose visual landmarks and their sensory features in order to describe and discriminate objects, and how to create short or efficient sequences of such descriptions for low cost (however defined) qualitative navigation from a given place to another.

2 Formal Definitions

2.1 The World, The Map-Maker, and The Navigator

The world we model is two-dimensional in the sense that objects are considered to be equivalent to their appearance, from a fixed vertical angle, and within a narrow, and therefore orthographic, field of view. In that sense, it is very similar to the world as it would be displayed in an aerial photograph, if that photograph has no symbolic information superimposed upon it. Since the photograph has not been taken at an oblique angle to the ground plane, there are no significant occlusions. We further restrict the navigator's movement to be in a constant plane above and parallel to the ground plane, and the navigator's vision to be normal to it.

Relaxing the restriction of a constant plane would complicate the giving of directions, since viewing height would be variable, would have to be specified by the direction-giver, and introduces an additional degree of

freedom over which the choice of landmarks would have to be optimized since most landmarks are only landmarks relative to their visible surround. Relaxing the restriction of normal view would be more critical, since it introduces two degrees of freedom and introduces visual occlusions. In principle, the possibility of occlusion deeply impacts the definition and detection of landmarks, since at least part of a landmark's effectiveness comes from its visibility over long ranges. We restrict ourselves now, however, to visual navigation with two degrees of freedom over a two-dimensional world.

We find it critical to distinguish three similar but subtly different perceptions of "the world".

The two-dimensional world as it exists and is experienced by both map-maker and navigator is mathematically rich: it is continuous, it has a distance measure, and objects "embedded" in it can have finite extent. Much of this appears to be extraneous: many such worlds consist largely of the essentially empty space between objects, much navigation can be done by simple object order rather than object distance, and objects can often be considered to be point-like.

We therefore postulate that the map-maker, omniscient and error-free, has abstracted the world into something more akin to a planar graph. That is, the world is conceived as a collection of nodes which represent significant objects, connected by arcs which represent navigable regions without intervening objects. Empty space, distance, and object extent are ignored. The omniscience of the map-maker ignores the very difficult issues of map induction (see [Dean, 1987]), and the infallibility dodges the issues of partial or errorful information, deliberate camouflage, unexpected events, or even the proper navigator strategies for "believing" a map and for recovering from any errors. However, to our map-maker, a plane flight would be simply the graph (or subgraph), (*New York*)-(London)-(Paris)-(Rome).

The map-maker communicates even less of this subgraph to the navigator in the form of a sequence: this is, the best, by some criterion, "custom map" of visual landmarks, ordered by sequence of traversal but not ordered by space, or even by actual time of encounter. In fact, the actual navigation path can be self-intersecting; in some cases, in order to be optimal, it must be. It is this second level of abstraction and the process of its creation are the foci of the paper. Throughout we assume that there is only one navigator, and that the map-maker's omniscience extends to a perfect model of the navigator's sensory endowment.

The navigator perceives the world in the most limited way. Sensory range is limited, perhaps only to the current object being experienced, and there is virtually no memory. However, this is an actual advantage, usually the less that needs to be sensed or remembered the better, since the cognitive load of the navigator is minimized. Although more intelligent navigators can be modeled, this one chooses to ignore most of the world and its features for the sake of efficient traversal.

For consistency, we will refer to the abstraction of the world as the "world model" or the "planar graph", and the custom map as, simply, the "map". In layman's

terms, the world model is itself usually called a "map", particularly if it is not drawn to scale, and the custom map would probably be called "the list of directions." However, in some instances, particularly at car rental agencies equipped with electronic custom map-makers, the layman can be given exactly what we call a map, if he has a unique destination in mind. Surprisingly, recent research shows that maps as we define them are preferred by human beings over the Exxon-variety world models [Streeter *et al.*, 1985].

2.2 Landmarks and Custom Maps

To consider objects as point-like requires enough processing intelligence in the navigator to recognize an object as a single object, and to capture, hold, and dismiss the single object from its sensory array; in effect, the navigator can "debounce". This is an important consideration, since the "experience" of an object must be a unified whole, even if the prior and succeeding experiences are identical. Practically speaking, this limits the amount of blind travel between objects; further, it defines what is meant by a single object from the point of view of the navigator. Three identical trees as seen from above must give rise to the experience of "tree" exactly three times, even though the trees are sensed at some distance, have spatial extent, and may overlap, and are still sensible after some distance, often even when the next tree is also sensible.

Given the model of unified, immediate perception, which implies that interframe time is small, objects can now be modeled by the map-maker by some symbolic abstraction. This object model must be communicable to the navigator, and the navigator must be capable of modeling its own perception in these terms. It is clear that the grammars for describing objects are many, but they are assumed here; this paper is more concerned with inter-object distinctions and relationships. The definition of a landmark ultimately invokes the ability of a navigator to distinguish a landmark from any other object in its graph neighborhood; a landmark is an object that can be recognized by a navigator along a path through the world graph, regardless of the intervening object nodes. Thus a tree in a desert is landmark, but a tree in a forest is not. Landmarks can be relative, or even "one-sided" - that is, path-dependent; for example, a tree on a forest's edge is a landmark only on the way in. We will make the definition of landmark more precise shortly, at least for our level helicopter world.

A custom map is a sequence of directions; each direction is a pair consisting of a heading and a landmark description. The navigational goal is simply the last landmark description. The first item of the pair, the heading, can be quite general, as it can specify a strategy (*wander northward, spiral outward*), as well as an absolute heading (*east*) or relative heading (*turn left*). In extreme cases, the navigable regions between landmarks are capable of being continuously sensed as a series of landmarks, e.g., a road and its sides, and the heading can be specified in terms not addressed in this paper, e.g., follow the road. The second item of the pair, the landmark description, can be arbitrarily complex, not

only with respect to the need to efficiently transmit an object model, but also with respect to any spatial grammar superimposed on object primitives, e.g., "three occurrences of big houses next to things other than a big tree or a small pole". The theory of LR(k) parsers addresses some of these issues, since basically a direction based on a compound object is a simple sensory program.

We note as a side issue that our world ignores those "reassurance" directives that are common in human map-making, e.g., "along the way you will pass a W", or, "if you see an X you have gone too far, turn around and look for the Y." Their purpose seems to be twofold, and both are ignored in our world. The first purpose is, they calibrate the navigator during an interactive direction-giving (for instance, the navigator can ask "What's a W?") but our direction-giving is not interactive. Secondly, they serve as a warning that a neighborhood is becoming visually confusing, and that more precise sensing is called for, (effectively, Scan for X-confusable objects); our navigator has no sensory choices.

2.3 Navigating with a Custom Map

Having defined what a custom map is, we now examine its use. Then we will address its creation, which is one of the central concerns of this paper.

The most primitive decision a navigator must make is whether or not its observation of the world is compatible with some object description given by the map-maker. We model such an operation by a boolean-valued function "match", which takes as input a description, and uses the description to schedule the appropriate sensory and cognitive information to make the judgment.

To illustrate navigation, consider the following world model, which is impoverished to the point of being linear. Objects are described by two point-like features, color and size, taken from the domains red, green and small, large, respectively. There is no spatial relations in the world other than forward and backward (" $+$ " / " $-$ "), and there is no intra-object spatial relations at all. The world is given by the graph $(gs)-(rs)-(gl)-(rl)-(gs)-(rl)-(rs)-(rs)-(gl)$, which is a straightforward syntactic simplification of the full definition of $(green, small)-(red, small)-\dots$. Directions such as $+rl$, or $-gs$, or even $+l$ tell the navigator to move forward until the red large object is sensed, or backward to the green small one, or forward to the large one, respectively. The last direction is a partial object description, and the ability to partially describe landmarks greatly complicates the selection of landmarks and their descriptions, although it likewise greatly simplifies navigation itself.

The navigator requests a custom map to traverse from the first object to the next to last one. The epistemological issues of how a navigator knows and/or communicates both present and desired positions to the map-maker are side-stepped here. The custom map maker can reply with over 300 custom maps; among them are $+s+s+s+s, +g+gl-r, +gs+s+r, +lr+sr+r, +g+r+g+r+g+r+r+r$. The first map uses only one object model. The second causes overshooting of the goal, but has only three directions, and therefore minimizes communication time. The last map is one of many

that minimizes distance, but enumerates all objects between the start and the goal. Depending on the way in which sensory costs are tallied, the third or fourth may minimize sensing. Even in one-dimension, there is great variety possible.

In two dimensions, given a vocabulary of headings, and a vocabulary of object models and their allowable partial models, the number of potential custom maps grows exponentially. Thus, we would like to make the idea of "efficient" and "best" map precise. Unfortunately, we first need some additional definitions and representations to investigate how difficult these questions are; we present them in the context of the one-dimensional world example. Their extension to two-dimensional space is conceptually straightforward, but computationally daunting.

2.4 Transition and Cost Graphs

We introduce two abstract data types, the transition graph and the cost graph. They are abstract data types because, depending on the vocabularies for headings and object models, the map criteria being optimized and, in some cases, the contents of the world model itself, they are realized as various data structures. Thus, in some cases they are best implemented as arrays, in others as lists or trees. In some cases they are best established before the optimizing map-making algorithm, in others they are incrementally established cooperatively with it, and in still others they are never established at all, but are directly incorporated into the algorithm itself.

Intuitively, the graphs record information about the most efficient direction (heading plus object description), if any, that takes a navigator from any given node in the world model graph to any other node. If there is such a unit direction, the transition graph records it, and the cost graph records its cost. This cost can be defined in multiple ways: it can be the distance traveled, or the amount of sensing necessary to tell that intervening objects are not the goal, or simply the complexity of the heading strategy. Both graphs conceptually contain a special sentinel wherever a unit transition is impossible, which is usually the case; in particular, there is no unit direction that can cause the navigator to remain in the same place.

The problem of obtaining an efficient custom map from a start to a goal can then be broken down - conceptually, at least - into the establishment of efficient unit directions, followed by the traversal of the unit direction graph from start to goal. There are times, of course, when this is grossly inefficient, such as when the navigator is traveling a very short distance and a type of search is more appropriate; we do not thoroughly analyze such tradeoffs here.

Intuitively, the graphs are filled in in two stages.

First Stage; Formal Definition of "Landmark"

In the first stage the transition graph is filled with tentative unit directions of the most inefficient kind, namely the full object description of the unit direction's goal. For example, in the example one-dimensional world model $(gs)-(rs)-(gl)-\dots$, the unit transition connecting nodes 1 and 3 would initially be given as $+gl$,

even though either $+g$ or $+l$ would suffice. In the most general case in two dimensions, creating the world planar graph, and labeling the arcs between two reachable nodes is a complex procedure, highly dependent on the vocabulary of headings and objects, and it may be non-trivial to select the unit directions between nodes in an efficient manner. In special cases, such as the one-dimensional world given above, or in the cartesian two-dimensional world actually implemented, clever encodings do yield good efficiencies of both processing and storage.

For the example, the full first stage transition graph is shown in Table 1, where empty entries are considered to be filled with sentinel values.

	gs	rs	gl	rl	gs	rl	rs	rs	gl
gs		$+rs$	$+gl$	$+rl$	$+gs$				
rs	$-gs$		$+gl$	$+rl$	$+gs$	$+rs$			
gl	$-gs$	$-rs$		$+rl$	$+gs$				$+gl$
rl	$-gs$	$-rs$	$-gl$		$+gs$	$+rl$	$+rs$		$+gl$
gs	$-gs$	$-rs$	$-gl$	$-rl$		$+rl$	$+rs$		$+gl$
rl		$-rs$	$-gl$	$-rl$	$-gs$		$+rs$		$+gl$
rs		$-rs$	$-gl$	$-rl$	$-gs$	$-rl$		$+rs$	$+gl$
gl			$-gl$		$-gs$	$-rl$	$-rs$		$+gl$

Table 1: First stage transition matrix

In the most general case, the transition graph will be indexed by node number, and the entry for a unit transition between node i and node j will record the heading in its vocabulary and the full object description of j . However, what is most significant is that in general, if navigation is in-the-large, by definition there will be a large number of objects with the same object descriptions; since only the first (that is, the nearest) object is attainable by a given unit direction, the transition graph will always be sparse, and it may be compressible by the usual techniques.

We now note that one definition of a "landmark" would be an object whose column in the transition graph is very nearly filled, that is, an object that can be obtained under a unit direction from very nearly everywhere. In the above example, the gl object is an optimal landmark. One can easily define a concept such as the "landmark radius" of an object in the obvious way. Because landmarks can be used to navigate over great numbers of objects, they will tend to appear regularly in custom maps. However, landmarks need not be "reversible"; for example, the fourth object in the above example is the first of two such objects close together, and is only valuable as a landmark while navigating to the right; it is perceptually "shadowed" by the sixth object when navigating to the left. In other words, traveling left from most places it is impossible to reach it in one step.

Second Stage In the second stage of transition and cost graph construction, the transition graph is optimized by selecting the minimal object description that maintains object uniqueness. The cost of this optimal unit direction is recorded in the cost graph, sentinel values in the transition graph giving rise to infinities in the cost graph.

In general, the second stage is more complex, conceptually and computationally, for several reasons. First,

there may be more than one optimal partial object model that attains the optimal cost: in the example, the unit direction from 1 to 3 can be $+g$, or $+l$. Depending on the overall model of map goodness they may all have to be recorded. Secondly, depending on regularities in the world, or the vocabularies of the navigator directions, the optimization of the entire graph may probably be done in a more efficient way than by optimizing each entry separately. But third and most importantly, the optimization of even a single unit transition is NP-complete in the number of partial models. Selecting the optimal partial model can be obtained by heuristic search, with the most effective heuristic appearing to be the one that says "select the partial model that conflicts most with all the other objects along the path to node j " [Kender and Leff, 1989]. In short, the topological visual navigation problem subsumes the problem of minimal object description, except that it provides a well-defined context: the context of the path to the object.

Applying optimization to the example, we find the following full second stage transition and cost graphs, in Table 2 and Table 3, where cost is given as sensing cost over the intervening objects.

2.5 Custom Map Creation

At this point, the problem is no longer a vision or a robotics one; standard graph search algorithms such as Dijkstra or A^* over the transition and cost graph can be used to find the least costly path from a given node to another. The resulting sequence of unit directions is the custom map.

3 Application to the Level Helicopter World

3.1 The World Model

The initial application of these ideas is to an highly idealized two-dimensional world. Nevertheless, it serves as an arena for the exploration of significant computational and representational problems. The world is divided into M rows and N columns, and the navigator's vocabulary of headings is restricted to traveling in one of the four directions: *north*, *east*, *south*, and *west*. Although severely restricted, this world still adequately models much navigation in urban areas, like taxi-cabs in New York, or within man-made structures, like mail delivery in offices. Restricting the world to rows and columns still leaves room for a larger vocabulary of heading strategies: various space-filling searches, for example.

An object, or interchangeably a landmark, can be placed in any cell of this grid. Given current limits on processing speed, we define as our object model a 3×3 matrix of dots; this allows 511 possible objects, but far fewer if rotation of objects are not considered distinct. There are no object symmetries due to rotations in this world, since the camera is room-oriented, not path-oriented: that is, as our robot arm changes direction, the camera does not rotate. (This is roughly equivalent to a human being's navigating by turning his road map on his lap as his car turns.) However, all objects must

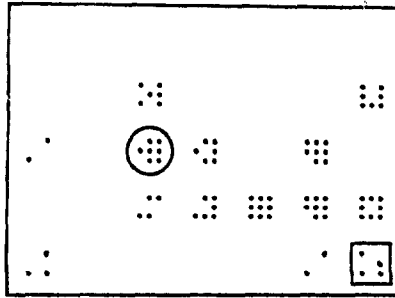


Figure 1: Sample dots world

have at least one dot on each of the four edges of the matrix. This simplifies the *adjust* phase of the navigation, and avoids the epistemological issues such as how many objects there are that have a single matrix dot (one or nine?). Figure 1 shows an example of the map-maker's view of our experimental world of size 7×5 . The landmark enclosed by the square is the start point and the landmark enclosed by the circle is the goal.

Because there are no occlusions, objects appear the same from all viewing directions; this simplifies custom map creation. In general, however, object models must address the characteristic view problem, the occlusion problem, and simultaneously be flexible enough to accommodate multiple kinds of descriptive features that the sensors of the navigator can identify: size, color, texture, shape, number of components, topological relations between components, etc.

3.2 Navigator model

We used a CCD array camera mounted vertically on a robot arm as the navigator's means of observing its environment. The height of camera was fixed above the world model and it could only move in a plane parallel to it. This obviated collision detection algorithms. Because viewing was normal to the ground plane and fixed, there were no occlusions nor even choices of field of view. Field of view was a fixed window, and very small relative to the navigational environment. At most two landmarks were visible to the navigator at any instant; nearly always there was a fraction of one or less.

The navigator is capable of operating in two different phases: *seek phase*, and *adjust phase*. It is assumed to have no metric capabilities outside its view window, and within its view window it perceives by a type of symmetry calculation. During seeking, the navigator continuously looks for any object coming into its viewing window. When an object is spotted on the boundary of the window, the navigator adjusts its position so that the object is centered in the view window. Adjustment not only counteracts errorful drift; it may be strictly necessary on turns to properly align the next heading. Other than seeking and adjusting, the navigator is not intelligent and makes no decisions; the map-maker has made them all. (The opposite extreme is that of a navigator

with no map at all: everything depends on the intelligence of the navigator's search.) Navigation succeeds if the navigator attains the final object description.

3.3 Map-maker model

In the level helicopter world, each direction in the custom map has the format (*heading, description*). Headings are the four compass points, and the partial object description is simply the leading edge of the 3-by-3 matrix of dots, in the direction of travel. For example, (*north, '...'*), (*west, '.'*) is interpreted as travel north until you see three dots in a row, then move travel west until you see a single dot on the right (i.e., the north). There does not appear to be any advantage in encoding the description further, either in the custom map, or in any of the intermediate data structures used by the map-maker. The object itself, though impoverished, has four different views of three tokens each, and they are trivial to obtain; the object itself has much less. Depending on the ease with which characteristic views can be obtained from objects in general, it may happen that the best description of an object is the object itself. The computational cost of obtaining and storing all views, most of which by definition will not be visible or useful in navigation-in-the-large would be even more excessive than in the special case here.

3.4 Definitions of custom map quality

Since there are typically multiple custom maps to travel between any two landmarks in the grid world, it is necessary to define a quantitative model for evaluating the quality of a custom map. There are a number of metrics to measure the cost of a path. Each metric may lead to a distinct path, so each metric may lead to its own heuristics. The choice of metrics depends on the goals of the map-maker and the navigator, and on the characteristics of the grid world. Two possible goals are minimizing travel time and maximizing the probability of reaching the goal.

One obvious cost metric is distance, measured by the Manhattan metric. A second is sensor cost. In our model, this is the number of times along the path that a dot is detected, which reflects the number of times the navigator needs to stop and check if it has reached the next specified object in its set of instructions. In general, the robot may have different sensors with different costs, or the cost to verify an object may vary depending on the landmark.

Time of travel is a third possible metric. Relevant data would include speed limits, obstructions, and sensing times for the available landmarks. Speed limits are generally dictated by law or by roughness of the terrain. The average waiting time for certain obstructions, like traffic lights, may vary depending on the direction of travel. A fourth metric is map length, which is the number of directions that the map-maker gives to the navigator to get from one point to another. This metric measures map transmission costs and memory requirements. A fifth gives preference to straight paths; Elliott and Lesk [Elliott and Lesk, 1982] noted that the number of turns and the direction of the turn also affect the time

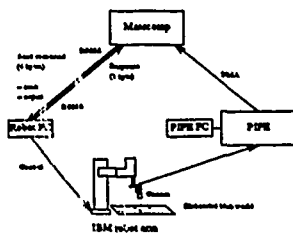


Figure 2: Configuration of the experimental equipment

of travel, and therefore counted right turns as 1/8 mile and left turns as 1/4 mile.

All of these metrics tradeoff with reliability, which generally decreases with distance traveled. This suggests a class of non-monotonic reliability metrics, which would penalize both very short and very long unit direction distances. A second class counts as its cost some function of the confusability of the objects along or nearby the specified path: some objects are not only unique, they are unique in robust ways. A third class gives preference to "rear view mirror" navigation, that is, prefers landmarks that are distinctive on both sides, so that landmarks can be verified once obtained.

Of course, all of these metrics can be combined in numerous ways. And some or all of these metrics may cause the navigator to back up. For example, suppose the navigator is due west of the destination, all landmarks between the navigator and the destination, including the destination landmark, are red, and a green landmark is found due east of the destination. It is easier to specify the instruction to the destination as "go east until you reach the green landmark, then go west to the red landmark."

4 Implementation

The world, map-maker, and navigator were implemented. A Masscomp [Masscomp, 1986] [Masscomp, 1988] hosted the map-maker, and the high level control of the navigator. Basic image processing was performed using a PIPE (Pipelined Image Processing Engine) system [Aspex, 1987]. The navigator was a camera with a 12.5mm lens mounted on an IBM 7575 robot arm [IBM, 1986], programmed using IBM's AML/2 [IBM, 1986] (Figure 2).

The logical components of our system consists of three main modules. The first, an environment creator, has a user-friendly interface for entering object descriptions and object positions. The second, the navigator module, reads the set of commands generated by the map-maker program and performs the navigation. During the seek phase, the leading edge of the image is monitored by the PIPE. Upon detection, it adjusts and attempts to verify object identity by its dot signature. Some calibration was necessary to ensure that during the monitoring

movement was not fast enough to "lose" a dot.

The third component, the map-maker, is the most complex. Much of its efficiency came from the exploitation of the observation that unit directions could only travel within rows or within columns of the world. Being in the same row/column is a necessary but not sufficient condition for an object to be reachable; it also could not be "shadowed" by an object that had an identical leading edge. Note, however, that reachability is not symmetric.

When every pair of landmarks has been examined for reachability, we can produce sets of connected components. We define such a set as *highway*. For a given world, multiple highways can exist. Thus, the navigator can travel from one landmark to another only if these two landmarks are on the same highway. Computing highways can be done in $O(M + N + K)$, where M and N are the row, column sizes of the grid world and K is the number of landmarks populated. In $O(M + N)$ we bucket sort the cells, once as (i, j) and once as (j, i) . Then we have only to find the edges between adjacent points in rows and columns. Since we have K landmarks, the result is a graph with K vertices and $O(K)$ edges; in $O(K)$ time we find its connected components. This implies that it takes $O(M + N + K)$ to find out whether the goal is a place that "you can't get to".

There are two main data structures used by the map-maker module, the transition matrix and the cost matrix. The entry (i, j) in the transition matrix is filled with a direction - north, east, south and west - and a descriptive feature value of the landmark j , if it is reachable from landmark i . For example, if j is at relative north from i and its south edge has value 7, i.e., three dots in a row, then the transition matrix entry (i, j) would be (north, 7). The cost matrix contains the cost of such a transition. In our model, we computed three different optimality criteria: sensing cost (s), distance traveled (d), and the map length (m). Distance traveled is defined as the number of cells the navigator travels over. Sensing cost is the number of landmarks encountered on the way. Map length is the number of instructions given to the navigator.

The map-maker computes the optimal route, based on the optimality criteria, by Dijkstra's shortest path algorithm. The time complexity for Dijkstra's algorithm is $O(n^2)$ if no fancy data structure is used [Aho *et al.*, 1974], but if a Fibonacci heap is used as a priority queue, Dijkstra's algorithm takes $O(e + \log n)$, where e is the number of edges and n is the number of vertices.

Another way of finding the optimal path is to use a heuristic search algorithm such as branch-and-bound or the A^* algorithm [Rich, 1983]. These algorithms are more costly than Dijkstra's algorithm in the worst case, but if the heuristic function is "good", they can be very effective. We found that in the case of sensing cost (s) estimation or map length (m) estimation, the underestimating heuristic that assigns a cost of 1 to objects in the same row/column, and 2 elsewhere occasionally gives dramatic improvement. Also in the case of distance cost (d) estimation, using the Manhattan distance metric between current position and the goal as underestimating heuristic gave us good results. We are attempting to

quantify under what circumstances heuristic A^* outperforms Dijkstra.

Note that optimal paths can be computed only if the start and goal landmarks are on the same highway. Inter-highway jumping methods are possible, but costly and possibly inaccurate. We have defined several *wandering strategies*, which are basically a space-filling zig-zag movement of the navigator towards a certain direction. The space that is filled can be rectangular with a specified width (a width of 1 gives standard navigational seeking), or triangular, which eventually fills a specified quarter of the plane, or semi-triangular, which fills an eighth.

Although it is difficult to analytically determine the expected number of highways given a specified object density, a series of experiments suggested that the number of highways is at a maximum when object density is about .03. Below .03, objects tend to be completely isolated; above .03, highways start to link up into one world-wide highway. An analytic proof would have some of the flavor of the proof of the birthday paradox: to start a new highway, each additional object would have to avoid all existing highways, which becomes increasingly difficult.

5 Experimental results

For our experiments, we have allowed one primary cost optimality criteria and one secondary criteria. Six different paths were possible: sensor-distance, sensor-maplength, distance-sensor, distance-maplength, maplength-sensor, and maplength-distance. The difficult part was coming up with a set of landmark placements so that the paths generated by the map-maker are as distinct as possible. There seemed to be no apparent heuristic to do so, although it is clear that $m \leq s \leq d$. Using the landmark layout in Figure 3, we have the following paths:

$d-s$: 11 8 9 10 6 3 ($d=6, s=5, m=5$)
 $d-m$: 11 12 2 3 ($d=6, s=6, m=3$)

 $s-d$: 11 12 13 4 3 ($s=4, d=8, m=4$)
 $s-m$: 11 0 1 3 ($s=4, d=10, m=3$)

 $m-d$: 11 12 2 3 ($m=3, d=6, s=6$)
 $m-s$: 11 0 1 3 ($m=3, d=10, s=4$)

Figure 4 shows two of the optimal paths ($s-m$ and $s-d$) generated by the map-maker. The numbers on the arcs indicate costs $d/s/m$.

The algorithm to derive optimal paths that satisfies the primary and the secondary criteria had to be carefully designed in order to prune the exponentially large number of paths created. There are several ways to solve this problem. The first is a n -stage search method, where n is the number of optimality criteria being used. Using the primary criteria, the algorithm creates a tree of primary optimal paths, then in the next round of search, considers only the edges that appeared on the primary paths. If we have n optimality criteria, we apply this n times. The second method uses n cost matrices, but computes the optimal paths in a single pass. The search is done using the primary cost matrix and when there is

	gs	rs	gl	rl	gs	rl	rs	gl
gs		+	+	+				
rs	-		+	+	+			
gl	-	-		+	+	+		+
rl	-	-	-		+	+	+	+
gs	-	-	-	-		+	+	+
rl		-	-		-		+	+
rs		-	-	-	-	-		+
gl			-	-	-	-	-	+

Table 2: Second stage transition matrix

	gs	rs	gl	rl	gs	rl	rs	gl
gs	1	2	3	6				
rs	2	1	1	2	6			8
gl	5	2	1	1	2	5		7
rl	6	3	2	1	1	2		4
gs		6	5	2	1	1		3
rl		7	6	2	1	1	1	2
rs			7	3	2	1		1
gl			8	4	3	1		1

Table 3: Cost matrix

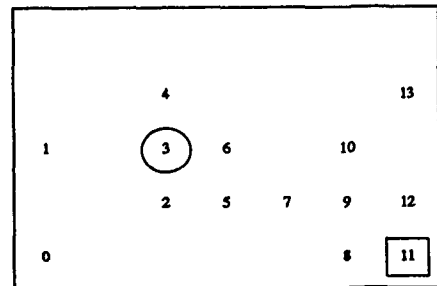


Figure 3: Labeled sample dots world

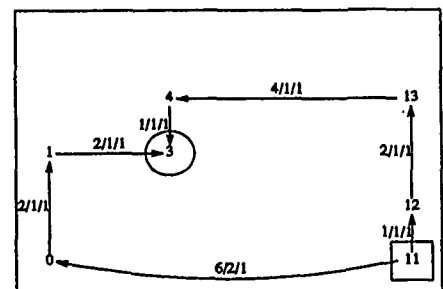


Figure 4: Paths generated by map-maker

a tie, the secondary cost matrices are considered. This second method increases matrix computation time and space to save search time. A third method uses a single compressed cost matrix and a single-pass. Each entry in the cost matrix is an algebraic combination of primary and secondary costs, encoded in odometric fashion. A single pass of search would suffice. However, one drawback of this method is that if the number of optimality criteria is very large, we would have to use a very large number of bytes (or words) to hold each cost matrix entry; further, the computation involving these numbers would also be costly.

6 Conclusion and future work

Navigation-in-the-large addresses some of the fundamental issues of AI in computer vision. Representations of objects with an premium on uniqueness, heuristics for finding good visual landmarks quickly, and the design of fall-back wandering scans of an environment all appear necessary. With the need to handle object occlusions or navigator oblique views, rules of thumb become paramount. We hope the end result of these investigations would be a greater understanding of how to quantify the visual difficulty of a world, and how to quickly determine what visual features or viewing strategies would provide effective custom maps for navigation.

References

- [Abbott, 1952] Edwin Abbott. *Flatland: A Romance of Many Dimensions*. Dover Publications, 1952.
- [Aho et al., 1974] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.
- [Aspex, 1987] Aspex Inc. *Introduction to PIPE Model 1 Systems*, December 1987.
- [Davis, 1986] Ernest Davis. *Representing and Acquiring Geographic Knowledge*. Morgan-Kaufmann, 1986.
- [Dean, 1987] Thomas Dean. A taxonomy of map learning problems. Technical Report CS-87-16, Department of Computer Science, Brown University, August 1987.
- [Elliott and Lesk, 1982] R. J. Elliott and M. E. Lesk. Route finding in street maps by computers and people. In *Proceedings of the 1982 AAAI Conference*, pages 258-261. IEEE, 1982.
- [Garey and Johnson, 1979] Michael R. Garey and David S. Johnson. *Computers and Intractability*. W. H. Freeman, 1979.
- [Gleason, Jr., 1979] H. A. Gleason, Jr. *Linguistics and English Grammar*. Holt, Rinehart, and Winston, NY, 1979.
- [Hanvey, 1988] Monnett Hanvey. Environmental representation for mobile robot navigation. Technical report, Department of Computer Science, Columbia University, January 1988.
- [IBMa, 1986] IBM. *IBM 7575 and 7576 Manufacturing Systems Hardware Library Site Preparation, Installation, and Specifications, Part Number 70X8867*, 1st edition, August 1986.
- [IBMb, 1986] IBM. *IBM 7575 and 7576 Manufacturing Systems Software Library AML/2 Manufacturing Control System User's Guide, Part Number 67X1370*, 1st edition, August 1986.
- [Karp, 1972] Richard C. Karp. *Reducibility Among Combinatorial Problems*, pages 85-103. Plenum Press, New York, NY, 1972.
- [Kender and Leff, 1989] John R. Kender and Avraham Leff. Why direction-giving is hard: The complexity of linear navigation by landmarks in one-dimensional navigation. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1656-1658, November/December 1989.
- [Kuipers, 1978] Benjamin Kuipers. Modeling spatial knowledge. *Cognitive Science*, 12:129-153, 1978.
- [Masscomp, 1986] Massachusetts Computer Corporation. *MC5600/5700 Installation Guide, Part Number 075-04007-00-0 (Revision A)*, April 1986.
- [Masscomp, 1988] Massachusetts Computer Corporation. *UNIX Programmer's Manual, Part Number 075-01012-00-0 (Revision K)*, February 1988.
- [Rich, 1983] Elaine Rich. *Artificial Intelligence*. McGraw-Hill, 1983.
- [Streeter et al., 1985] L. A. Streeter, D. Vitello, and S. W. Wonsiewicz. How to tell people where to go: Comparing navigational aids. *Int. J. Man-Machine Studies*, 22(5):549-562, May 1985.

Annotated Maps for Autonomous Land Vehicles

Charles Thorpe
Robotics Institute
Carnegie Mellon University

Jay Gowdy
Robotics Institute
Carnegie Mellon University

Abstract

Autonomous land vehicles need more information than is contained in standard maps. We have designed and built a new structure, the annotated map, to manage additional information. Annotations hold a wide variety of knowledge, both procedural (actions and methods) and declarative (data), tied to particular map locations and objects. Annotations can range from high-level ("circular object") to geometric ("position 10,15, radius 0.5") to sensor-specific ("possible position error .2") to raw data ("color R1 G1 B1"). The knowledge in an annotation can come from a wide variety of sources, such as human experts, mission planning software, and even the vehicle's own observations and experiences on previous missions.

A map manager module controls the annotated map. Two forms of access are provided, queries and triggers. Queries allow a module to fetch information on demand. They return all annotations of the requested type within a specified polygon. Triggers are a special form of annotations, monitored by the map manager. When the vehicle reaches the trigger's location, the map manager automatically sends a specified message to a named module. Triggers are set up during mission planning, and are used to wake up sleeping processes at specified locations or to alert a running module to a change in conditions.

Annotated maps are not designed to be a master control, but rather to serve as a scratchpad (for queries) and alarm clock (for triggers). Annotations have a standard format for header information, such as type and location. The format for the rest of the annotation is defined by the modules that post and retrieve the annotations, and does not need to be interpreted by the map manager.

Annotated maps provide a convenient framework for organizing knowledge. Tying the knowledge in annotations to particular locations in the map makes it possible to pre-plan difficult mission segments, and to retrieve that information efficiently during execution. This framework enables missions that would not otherwise be possible, due to real-time constraints and limits in processing and algorithmic power.

1 Introduction

1.1 Motivation

Much of the information that mobile robots need is tied directly to particular objects or locations. Maps, object models, and other data structures store useful information, but do not organize it in efficient and useful ways. We have built a new map-based knowledge representation, the "annotated map", to index information to the relevant object and locations. The annotations can be used for a wide variety of purposes: describing objects, providing hints for perception or control, or specifying particular actions to be taken. We have provided a query mechanism to retrieve annotations based on their map locations. We have also built "triggers", which cause a specified message to be delivered to a particular process when the vehicle reaches a given location in the map.

These annotated maps serve a crucial role in enabling missions that are otherwise beyond the reach of autonomous systems. Control descriptors allow mission planners to specify what the vehicle is to do at particular locations, reducing the need for onboard planning. Object descriptors contain detailed instructions of how to recognize a particular object, or contain the appearance of this object as seen by a particular sensor on a previous vehicle run. Such information greatly simplifies the problem of seeing and recognizing objects. Geometric queries enable the vehicle to focus its attention on objects in its vicinity, reducing database access and matching time. The trigger mechanism frees individual modules from having to track vehicle position, allowing them to devote their processing to the task at hand or to lie dormant until they receive their trigger message.

Annotated maps do not by themselves solve difficult problems of sensing, thinking, or control for autonomous vehicles. Their contribution is to provide a framework that makes it easy for other modules to cooperate in planning and executing a mission. Annotated maps thus fill a need that is common to many different vehicles, missions, and architectures.

Many analogous annotated maps exist for human use. Aeronautical navigation charts contain symbolic descriptions of routes (airways) and landmarks, and include annotations such as "road closed" and "code call letters of radio navigation beacons". The "Triptik" system¹, which includes annotations for "road conditions" ("construction", "speed check"), "road type" (interstate, two lane, etc.), general condi-

¹Triptik is a registered trademark of the American Automobile Association

- geometric: intersection angle 45 degrees
- vehicle-specific: turn with a circular arc of radius 15m
- raw data: steering wheel position left 1200 clicks

Knowledge must be carefully organized if it is to be useful. If the vehicle has to sort through all bits of information it has about every possible object, it will overshoot the intersection long before it has figured out how to recognize it or deduced that it was supposed to turn. It is far better to have information tied directly to the map, or automatically retrieved as needed. The landmark recognition module, for instance, is able to ask for a description of objects within its field of view, and retrieve the knowledge it needs to recognize them.

Organizing knowledge by tying it to a map is the heart of our "annotated maps". The annotations contain knowledge about particular objects, locations, or actions. Annotations come in one of two classes: descriptors and triggers. Descriptors are passive, and are retrieved by queries based on geometry and object type. A query for "all objects of type 'intersection' in this polygon" would return the annotation for the requested intersection, if it were in range. Triggers are active, firing when the vehicle reaches a particular location or crosses a certain line. A trigger will send a message to a particular module, such as "controller: start turning hard left in five more feet".

The knowledge in these annotations comes from many sources, including human experts, mission planning software, and even the vehicle's own observations and experiences on previous missions. It is both declarative (data) and procedural (methods and procedures). The level of the annotations depends partly on the vehicle's computational capabilities. Simple vehicles, in known environments, are able to execute simple pre-planned missions by having every object and action completely annotated at low levels. A more challenging environment, with more variation over time, may require higher-level symbolic descriptors in the map and more reasoning at run time. Practical missions will probably require a mix of levels of detail. Even a sophisticated vehicle may, for instance, decide to record the locations of specular reflections from a mailbox, and use those specularities as recognition cues. It may be much more difficult to reconstruct a 3-D model from the observed data, and to later predict the appearance from the model.

Figures 1 and 2 show a typical annotated map. Figure 1 shows a map of a suburban area, including about 0.7 km of road with two T intersections, and a variety of 3-D objects. Object information was collected using the ERIM laser range finder, and the road information was collected by using the inertial navigation system to provide accurate vehicle positions while we traversed the route. Figure 2 shows a detail of the first intersection, including the Navlab's position during a run and several triggers.

The goal of this run was to drive from a house near the beginning of the map to a specified house near the end. Annotations were added to the map to enable the Navlab to carry out this mission. There were annotations to set the speed appropriately: up to 3.0 m/s in straightaways and down to 0.5 m/s in intersections. Other annotations activated and deactivated the module that uses the laser range finder to correct vehicle position based on detected landmarks. Before every intersection there was an annotation that switched driving

control from a neural network vision program to a module that used knowledge from the map of the intersection structure and dead reckoning to traverse the intersection. Finally, there was an annotation at the end of the route that caused the vehicle to stop at the appropriate object. The route was successfully traversed autonomously.

In this run, and a variety of other runs, we have successfully used nine different types of trigger annotations:

- set speed
- dead reckon through intersection
- resume vision after intersection
- start landmark matching
- stop landmark matching
- stop at objects
- stop and start fast obstacle detection
- use vision through intersection
- switch perception modules

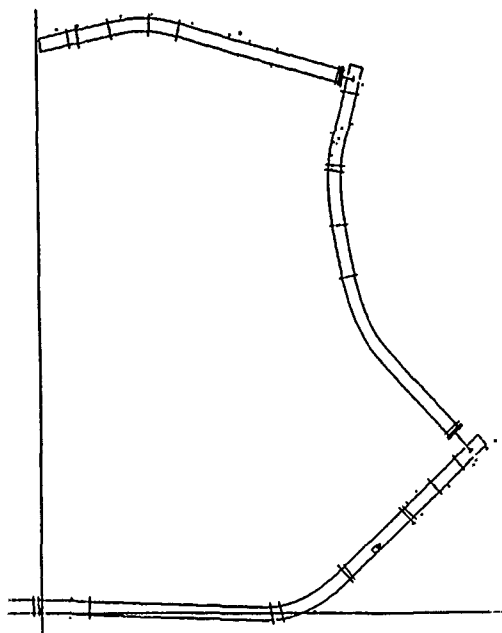


Figure 1: Map built of suburban streets and 3-D objects

3 Tenets of Map Construction and Use

Several key ideas underly our design for annotated maps, reflecting our experience in building perception and navigation systems for a variety of robots.

Minimize semantic interpretation. No-one can predict all the kinds of knowledge that will be placed in annotations. Moreover, the map module need not understand the annotations. The only common knowledge in annotations should be

tions ("winds through rolling hills"), points of interest (rest areas, gas, food, and lodging) etc. An intelligent human can usually drive a route without such aids; but they do provide a convenient framework for preplanning, and make "mission execution" easier. Furthermore, as we drive a route, we build our own mental representations of landmark appearance, curves in the road, and so forth, which we use to follow the same route more easily at a later time. Our annotated maps provide the same kind of functionality for autonomous mobile vehicles.

1.2 Related Work

At CMU, we have developed a family of autonomous mobile robots over the past ten years. Vehicles have included Neptune, a testbed for stereo vision and path planning [Thorpe, 1984]; the Terregator, our first outdoor mobile robot [Wallace *et al.*, 1985]; the AMBLER, a walking machine for planetary exploration [Bares *et al.*, 1989]; and, principally, the CMU Navlab [Thorpe, 1990, Thorpe *et al.*, 1988]. Our experience, especially with the Navlab, has driven the design of the annotated maps. We already have perception and control modules that can use information from annotated maps, including color vision [Crisman and Thorpe, 1990, Kluge and Thorpe, 1990], neural networks [Pomerleau, 1990], 3-D object recognition [Hebert *et al.*, 1990], and planning [Stentz, 1990]. We have also built the EDDIE architecture, which provides inter-module communications, control, and system structure for mobile robots [Stentz and Thorpe, 1989, Thorpe, 1989]. The tools provided by EDDIE will be used for the messages that underly queries and triggers in the annotated map.

Many other groups are working on related problems of mobile robots and knowledge representation. Rather than competing with the ideas of annotated maps, most of this research is providing useful tools and ideas that could use or help generate the annotated maps.

Fennema, Hanson, and Riseman at the University of Massachusetts are building world models and maps for their mobile robot, Harvey [Fennema *et al.*, 1989]. They have defined the concepts of "neighborhoods" and "locales". Neighborhoods divide space (and their map) hierarchically and topologically. Locales provide information for each neighborhood for the robot to determine whether or not the robot is inside that neighborhood. During planning, they generate a series of "milestones" and actions. Milestones are perceptual tests, performed to verify that a particular action has been accomplished. The UMass map and plan representations are similar to some of the uses of annotations, but have simple, fixed formats, are focused on declarative representations of 3-D object models, and do not provide map-based triggers. Also, the milestones and actions are generated for a particular plan, whereas annotated map triggers may include other kinds of actions that are not part of any planned mission, such as "if you ever enter this area, turn around immediately".

Rod Brooks at MIT has long argued for simple robots with simple control schemes and simple world maps [Brooks, 1986]. We concur that simple, sensor-based maps of particular locations are often useful. The lowest levels of our descriptor annotations are designed to contain precisely the sort of information that Brooks' robots use to calculate their position, or to cause a particular action, in a small local area.

We disagree with Brooks' contention that this is the only sort of information that a robot should remember. Robots often work in open, featureless environments, and need precise maps and accurate navigation even where no landmarks may be nearby. Annotated maps are designed to keep precise metric information in the geometric levels of annotations, as well as the lower-level cues advocated by Brooks.

Kender gives a much more abstract view of planning for sensor-based navigation [Kender and Leff, 1989]. He describes the combinatorial problem of deciding which sensors to use, and which landmarks should be recognized, in order to reach a given goal. The results of analyses such as Kender's should be entered into triggers, to tell the vehicle what to look for, and into object descriptors, to say how to look for those objects.

Blidberg and his associates at the University of New Hampshire's Marine Systems Engineering Laboratory have implemented world models for underwater mobile robots [Chappell, 1989]. Most of their work has concentrated on efficient descriptions of space, such as quadrees. These spatial descriptions are important, but do not include many of the other forms of knowledge (actions, descriptions) for which annotated maps are useful.

2 Scenario

A typical hypothetical mission for an autonomous land vehicle is mail delivery, which includes traversing a network of roads, and picking up and dropping off mail at various points. This mission involves following roads, recognizing and dealing with intersections, self-locating by finding given landmarks, and performing the correct actions at the correct places. As the vehicle approaches an intersection, it updates its position. Then it switches control modes from road-following to dead-reckoning through known intersection geometry. Tracking the roads requires using different strategies in different locations. The vehicle tracks an unmarked suburban road using a neural net road following [Pomerleau, 1990] or a road color classification algorithm [Crisman and Thorpe, 1990]. When it switches to a road with lane markers and stripes, it uses a feature tracking algorithm [Kluge and Thorpe, 1990]. As the vehicle approaches landmarks, the laser range finder module takes images and matches the observed landmarks to the known landmarks to update the vehicle position [Hebert *et al.*, 1990]. The vehicle can stop at a mailbox, and another module can perform an action.

Planning and executing this mission requires several types of knowledge: what to look for, and how to see it; what to do, and how to accomplish it; where to go, and how to get there. The knowledge may range from high-level symbols, to low-level raw data. Approaching the intersection, for instance, the perceptual knowledge would include:

- symbolic: intersection
- geometric: size and shapes of roads intersecting here
- sensor-specific: use laser range finder to pinpoint the position
- raw data: Object 2 meters tall, 0.4 meters wide at position (x,y)

Control knowledge can also span a range of levels:

- symbolic: turn left at intersection

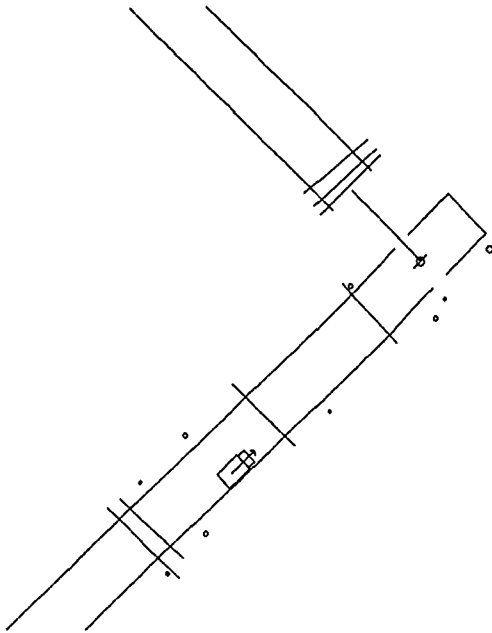


Figure 2: Trigger annotations for sensing and vehicle control

enough header information to store and retrieve the annotation. All the rest of the annotation belongs to the modules that create it and interpret it, to be decided upon by their creators. The annotated map serves only as a scratchpad.

No specialized query language is needed. The standard queries ask for all objects of type X within polygon Y. Any query more ambitious than that need not be supported. If, for efficiency, it is desired to have a more powerful query, it is best programmed in C in the map module, rather than inventing some other query language, translating queries into that language, and interpreting those queries in the map module. Answering queries for objects of type "landmark" and size "greater than 100", for instance, requires internal knowledge of the "landmark" annotations. This query should therefore be treated as a special case, only to be implemented jointly by the "landmark" annotation creator and the map module maintainer, and only to meet extreme efficiency requirements.

Separate global position tracking from local servoing. Maintaining the current position estimate in local coordinates is a real-time job, and is best done by the low level real-time controller. The controller's local coordinates should never be updated, so locations stored in local coordinates are always consistent. Commanded arcs, current positions of obstacles to be avoided, and other phenomena that are used once and then discarded, should be kept in local coordinates and never entered into the map. Map-based reasoning, such as matching landmarks against a map, or interpreting a position fix, are aperiodic events best done by a separate Navigator module. The Navigator maintains the transform from local to world coordinates. Any module that needs to know current vehicle position in world coordinates must acquire the Navigator's

transform, then apply that to the running position reports of the controller. In practice, acquiring the Navlab's current transform is done in one of two ways, specified at start-up:

- The Navigator can send its transform every time it is updated. This is used by fast-running modules that always need the latest update.
- Slower modules, that have a longer cycle time, may not need every updated transform. Worse, receiving too many updates before the module is ready to read them may cause the input queue to overflow. Instead, these modules are notified that a new transform is ready, but do not receive the update until they request it. The Navigator stores which modules have been notified and have not yet requested updates, to avoid sending repeated notifications.

Centralize position tracking. Modules often want to perform specific actions when the vehicle arrives at particular locations in the map. If each module were to continuously poll the Navigator and controller for current position, the controller could become overloaded. Active polling also means that those modules are using computer cycles. Moreover, a Navigator position update may skip the vehicle position estimate past the point for which a module is waiting. For each update, each module would have to figure out if any of its target positions had been passed. We prefer to have a single module, the map manager, doing position tracking for all modules. On reaching the points of interest, it awakens or signals the appropriate module. This is the function of "trigger" annotations.

No master control. The map module is best thought of as an alarm clock (for the triggers) and a scratchpad (for descriptors and trigger messages). It is not some "master" module that controls all thinking, and that therefore can become a major bottleneck. We prefer point-to-point communication between modules, with flow of data and control decided on module by module, rather than forcing all information through a single controller.

Plan incrementally. The map module is designed to be used by many programs, for many purposes, at many times. Some information may be permanent; other annotations may be added to provide directions for only a single mission. It is an advantage to be able to update, add, and delete at various times. In particular, display and user interface modules may read the annotated map from a file, look at it, display the annotations, change things, and write it back out.

4 Implementation of Annotations

The annotated map needs to provide efficient access, indexed by position. The annotations themselves need to contain an arbitrary amount of data, with a minimum of externally imposed organization on the contents. We have designed and implemented a two-part representation, consisting of a map grid and an annotation database. Each square of the grid contains a list of any annotations that are included in that square's area.

Adding an annotation to the map is a two-step process. First, the actual annotation is added to the annotation database. Secondly, the map grid must be updated. The location of the annotation is either a point, a line, or a polygon. This location can either be specified directly, for those annotations tied to

a location, or retrieved from an object description, for those annotations that describe an object. The location is then scan-converted (converted to a list of cells) into the grid, and a pointer to the appropriate entry in the database is written into the corresponding grid cells.

Retrieval of annotations in response to a query is also a two-step process. Queries can specify a polygon and an annotation type. The query polygon is scan-converted into grid cells. The annotations pointed to by each of those cells are collected, checked to see if they match the specified type, and returned.

Triggers work similarly. At each cycle, the map module calculates the current vehicle position. It calculates the line on which the vehicle has moved since the last cycle, and scan converts that line into the grid. Each cell through which the vehicle has moved is checked for trigger annotations. If any are found that have not already been fired, their messages are sent to their destination modules. Since the location of a trigger can be a point, line, or set of lines, a trigger can be fired when the vehicle reaches a certain location or when it enters a given polygon.

4.1 Representing Annotations

Annotations are represented with a uniform header format, plus a free-format data field. Typical header fields include:

header: type, destination module, used flag, text description, location, next object, previous object, data size
data: pointer to data

The header portion contains all the information that the map module needs to understand. "Type" and "location" are sufficient for answering queries; "destination" is required for sending trigger messages. The "used" flag is set when a trigger is fired, to avoid firing the same trigger repeatedly if the vehicle stays in the area covered by the trigger for more than one cycle. "Text description" is used by graphics display modules. This information is also sent as part of messages, to make it easier to debug receiving modules. The "location" of the annotation is used both in initially setting up the grid pointers, and for the use of the receiving module. "Next object" and "previous object" are used to describe extended linear objects. Extended objects may also have branches, which meet at intersections. Intersections have a center point, and any number of vertices, each of which points to the beginning of an extended object. The most common extended objects are roads, which are represented as short segments pointing to their preceding or following segments, or pointing to intersections.

The data portion of the annotation is, in the view of the map, an undifferentiated field of bytes. Any internal structure need only be understood by the modules that create and read the annotation. Since the headers have a known, fixed size, they can be stored in a random-access file. The data may be stored as a stream of bytes, with the header containing only a pointer to the beginning of the data and the number of bytes.

4.2 Implementation Details

Our prototype implementation has tested some of our design decisions, while other details will be decided after further data collection and analysis.

Grid cell size. If grid cells are too small, queries will have to look at large numbers of cells, and map storage will become a problem. But the querying becomes simpler, because any object found in any of the cells can be returned. Larger cells

give faster lookups, but are no longer selective enough to answer queries on their own. Instead, objects within grid cells must still be checked to make sure they are within the query polygon. For autonomous land vehicles with sensor ranges of two to thirty meters, a grid with 0.5 to 1.0 meter cell spacing probably provides the right tradeoff; our current implementation uses 0.5 meter cells.

Handling large maps. For a grid with 1.0 m cells, each square kilometer will contain a million cells. Each cell can be represented with at most a few bytes of data, depending on annotation density. The amount of memory required by a grid this size is easily within the capability of today's computer systems, but for missions spanning several kilometers, we will not be able to keep the whole grid in main memory at once. One possible solution is implementing quad-trees to take advantage of sparse data requirements over most of the grid. A more likely strategy is to keep the grid on secondary storage, and only keep a window around the current vehicle position in main memory. The annotation databases themselves may also need to be kept on backing store, and only read in as needed.

Distributed databases. Object descriptions might be most easily implemented in separate databases, internal to the modules that use them. Then the annotations need only return the index of the database entry. The problem with this method is ensuring consistency between databases in the modules, and indices in the grid. At the opposite extreme, the map annotations could contain all the data. The disadvantage of this approach is requiring more traffic between maps and objects. An intermediate approach is to start with all the knowledge in the map annotations, but have it automatically replicated in the appropriate modules at system initialization time. This ensures consistency while reducing runtime overhead, at the expense of startup costs. The design of distributed databases interacts with the design for handling large maps. Keeping annotations in individual modules would decrease the amount of information needed by the map module, and thus make building large maps somewhat easier.

In the current implementation, the annotation database is static during a run. When the system is initialized, the user adds stop points, turn points, or other triggers to specify the current mission. When the user is ready, the interface module saves the current annotation database and sends the name of the file to the map module. At start up, each module that needs a copy of the annotation database requests the name of the file from the map module. So modules contain a complete, consistent copy of the annotation database. The map module builds the grid, so it can handle geometric queries. It communicates with the other modules by specifying the index in the annotation database of the objects that match the current query. The map module also watches the grid for triggers.

Map update. Changing an annotation during a run would be no problem. Moving objects and annotations is more difficult. If a single object moves, it is easy to erase it from one part of the map and write it into another location. But if an entire portion of the map moves, such as discovering that a portion of the road is really longer than previously thought, the changes can be very hard to handle. Many objects would have to move: the road, all objects attached to it, all landmarks that were seen on previous inaccurate runs and indexed to the road, planned mission steps based on following

the road or on seeing those landmarks, etc. It is probably better to note the new information, keep running with the flawed map, and build a new map at the end of this run, rather than try to do updates on the fly. Map update strategy is also influenced by the "large maps" and "distributed databases" design issues. If an individual module updates its copy of an object description annotation, it will need to make sure any permanent information is written out when the run is terminated or when that portion of the map is overwritten by a new data window.

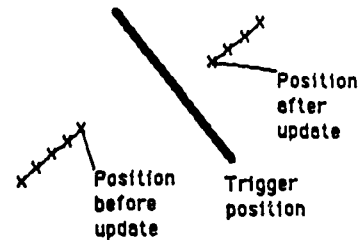
Since in the current implementation, each module keeps its own internal copy of the annotation database, map updates must be specially handled while building a new map. Under most circumstances, the map updates refer to objects that the vehicle will not see again on this run, and therefore the updates need not be propagated to all the modules. At the end of a run, all the new objects can be written to a new map file, to be used on succeeding runs. The exception is for building maps of intersections. Our procedure is to drive through the intersection, following one branch, and building a map; then to reposition the vehicle before the intersection, and follow the second branch. In order to register the two branches correctly, the perception and matching systems need to find newly-mapped landmarks. The map manager writes the annotation database to a file and notifies the relevant modules, which read in the updated database.

Interfaces. Conceptually, it is easy to add annotations to the map. A program reads in the annotation database, adds new annotations, and writes the updated files. Machine-generated annotations, such as object descriptions, use interface routines to read and write the map, and to insert annotations into the annotation database. Annotations added by hand require, besides the basic map interface routines, a user interface to point to locations or objects on the map, type or read the annotation data, display the resulting map, and ask for verification. While the format and contents of the annotations will vary, there is still a large body of common functions that use standard modules. We have built an interface, using X windows, that allows a user to add new objects and triggers to the map. The same interface is also used to display the vehicle and map during a run.

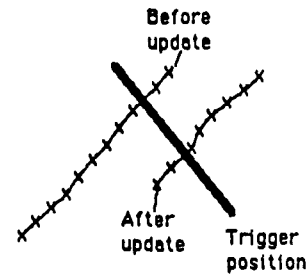
4.3 Trigger Details

In order for the map module to track vehicle position, it must know both the controller's current local position estimate, and the navigator's transform that relates local to global coordinates. Position queries to our vehicle controllers are efficient, returning in less than 10 milliseconds. Our current implementation uses an efficient process for getting transforms from the navigator, by having the navigator send the transform each time it is updated. Since landmark sightings or position fixes are relatively infrequent, an event-driven transform update is much more efficient than polling.

When the navigator updates position, the map module has to pay special attention to triggers. It may be that the vehicle position estimate will jump forward, skipping some triggers; or it may be that it will move backwards, creating the potential for firing triggers that have already been fired (see Figure 3). If the position update is relatively small, it makes sense to use the line of vehicle travel, plus the "used" flag, to make sure that all appropriate triggers get fired once. If the update is



Case 1: position update skips over trigger



Case 2: position update causes retraversal of trigger

Figure 3: Problems with refiring or mission triggers

large, it may no longer make sense to fire triggers that should have been fired long ago; and it may make sense to refire triggers that were fired very prematurely. Details of these design decisions are yet to be worked out.

The mechanism of notifying a module of a trigger is by sending a message over a port. In the Unix² operating system, ports can be set up by broadcasting their address and listening to the net to find out who would like to talk to them. Once connected, ports appear as files, and can be read and written easily. A module can easily check if there are any bytes waiting on its trigger port. If not, it has not received a message, and can continue running. If so, it can read the message header, allocate the memory structure for the message, and read the appropriate number of bytes into its memory. A running module can periodically check to see if a message is waiting. A sleeping module can simply block on read, which will cause it to pause until data arrives. It is possible to set timers, so a module can wait until either a timer expires or a message arrives, whichever occurs first. It is also possible to have an incoming message generate an interrupt, so the module can be notified while running even without checking for incoming data.

5 Conclusion

Annotated maps provide a framework to organize knowledge storage and retrieval for autonomous mobile robots. The Navlab group at CMU, and other groups around the world, have many of the individual pieces of a complete system: sensing, sensor understanding, local trajectory planning, con-

²Unix is a trademark of Bell Labs

trol, and vehicles. These pieces in themselves are only sufficient to perform limited tasks. Integrating those components into an efficient system is one of the difficult remaining gaps. The annotated map helps fill that gap. By providing generic data handling, it allows diverse modules to communicate their specialized knowledge. By tying this knowledge to specific locations and objects, the annotated map provides a focus of attention, using an efficient grid structure to answer queries about specific parts of the map. And through the automatic triggers, the annotated map eliminates the need for individual modules to attend to vehicle position and map location. We have built our first prototype annotated map, interfaced several modules to it, and used it to store and retrieve data during real Navlab runs. We are currently addressing the issues of large maps, and continue to interface more modules and to use annotated maps to manage a wider variety of knowledge.

6 Acknowledgements

Our work with autonomous mobile robots, and the Navlab in particular, is done with a host of colleagues in the Robotics Institute and School of Computer Science at CMU. Our thanks especially to Takeo Kanade and William Whittaker, co-principal investigators; to Jill Crisman, Martial Hebert, Dean Pomerleau, Didier Aubert, and Karl Kluge, who built the perception modules that the annotated maps support; and to Jim Frazier, who keeps the Navlab running and happy. Martial Hebert, along with our colleagues Stan Dunn, Joe Cuschieri, and K. Ganesan of Florida Atlantic University, provided useful comments on early versions of the article. This research is sponsored in part by contracts from DARPA, titled "Perception for Outdoor Navigation" and "Development of an Integrated ALV System".

References

- [Bares *et al.*, 1989] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker. Ambler: An autonomous rover for planetary exploration. *IEEE Computer*, June 1989.
- [Brooks, 1986] R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1), 1986.
- [Chappell, 1989] Steven G. Chappell. A simple world model for an autonomous vehicle. In *Sixth International Symposium on Unmanned Untethered Submersible Technology*. Marine Systems Engineering Laboratory, University of New Hampshire, June 1989.
- [Crisman and Thorpe, 1990] Jill D. Crisman and Charles E. Thorpe. Color vision for road following. In Charles E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 2. Kluwer Academic Publishers, 1990.
- [Fennema *et al.*, 1989] Claude Fennema, Allen Hanson, and Edward Riseman. Towards autonomous mobile robot navigation. In *DARPA Image Understanding Workshop*. Morgan Kaufmann, May 1989.
- [Hebert *et al.*, 1990] Martial Hebert, InSo Kweon, and Takeo Kanade. 3-D vision techniques for autonomous vehicles. In Charles E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 8. Kluwer Academic Publishers, 1990.
- [Kender and Leff, 1989] J. R. Kender and A. Leff. Why direction-giving is hard: The complexity of linear navigation by landmarks in one-dimensional navigation. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6), November/December 1989.
- [Kluge and Thorpe, 1990] K. Kluge and C. Thorpe. Explicit models for robot road following. In Charles E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 3. Kluwer Academic Publishers, 1990.
- [Pomerleau, 1990] Dean A. Pomerleau. Neural network based autonomous navigation. In Charles E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 5. Kluwer Academic Publishers, 1990.
- [Stentz and Thorpe, 1989] A. Stentz and C. Thorpe. Against complex architectures. In *6th International Symposium on Unmanned Untethered Submersibles*, June 1989.
- [Stentz, 1990] Anthony Stentz. Multi-resolution constraint modeling for mobile robot planning. In Charles E. Thorpe, editor, *Vision and Navigation: The Carnegie Mellon Navlab*, chapter 11. Kluwer Academic Publishers, 1990.
- [Thorpe *et al.*, 1988] C. Thorpe, M. Hebert, T. Kanade, and S. Shafer. Vision and navigation for the Carnegie-Mellon Navlab. *IEEE PAMI*, 10(3), 1988.
- [Thorpe, 1984] Charles E. Thorpe. *FIDO: Vision and Navigation for a Robot Rover*. PhD thesis, Carnegie-Mellon University, December 1984.
- [Thorpe, 1989] Charles E. Thorpe. Outdoor visual navigation for autonomous robots. In T. Kanade, F. C. A. Groen, and L. O. Hertzberger, editors, *IAS-2*, Den Haag, the Netherlands, 1989. CIP-Gegevens Koninklijke Bibliotheek.
- [Thorpe, 1990] Charles E. Thorpe. *Vision and Navigation: The Carnegie Mellon Navlab*. Kluwer Academic Publishers, 1990.
- [Wallace *et al.*, 1985] R. Wallace, A. Stentz, C. Thorpe, H. Moravec, W. Whittaker, and T. Kanade. First results in robot road-following. In *Proc. IJCAI-85*, August 1985.

Experiments in Autonomous Navigation

Claude Fennema and Allen R. Hanson

Computer and Information Sciences Department

University of Massachusetts

Amherst, MA 01003

Abstract¹

The University of Massachusetts mobile robot project is investigating the problem of goal-oriented navigation of an autonomous robot vehicle through a partially modeled, unchanging environment which contains no unmodelled obstacles. This simplified environment provides a foundation for research in more complicated domains.

Perception, planning, and execution of actions are integrated into a reactive system which reasons about landmarks that should be perceived at various stages of task execution. Perceptual servoing uses correspondences between image features and expected landmark locations to ensure proper plan execution and to maintain the relationship between the system's internal model and the environment. This paper briefly describes this system and presents experimental results which demonstrate the efficacy of perceptual servoing.

1. Introduction

The UMass Mobile Robot project [FEN90] is investigating the problem of enabling a mobile automaton to navigate intelligently through indoor and outdoor environments. The experimental platform, called "Harvey", is a Denning Mobile Robotics vehicle ultimately intended to navigate through offices, hallways, and university grounds as it carries out commands such as "Fetch the book" or "Take this to Claude". The initial system development efforts and experiments focus on robust goal-oriented visually guided navigation through a partially-modeled, unchanging environment that does not contain any unmodelled obstacles. If robust autonomous navigation can be achieved in this restricted domain, then a variety of challenging problems can be considered as the constraints on the assumed knowledge about the environment are relaxed.

2. System Overview

The philosophy underlying the system is the use of sensory data in a 'verification' mode. The system reasons incrementally about the actions necessary to accomplish its long and short range goals and about what perceptions

should result from the execution of those actions. Actions are based entirely upon the system's internal model, which includes a geometrically specified environmental model as well as the robot's spatial relationship to this environment. When an action decision is made, sensor data is compared with expected perceptions as the action takes place. Differences from expectations may modify the execution of the action, and possibly short range or long range goals.

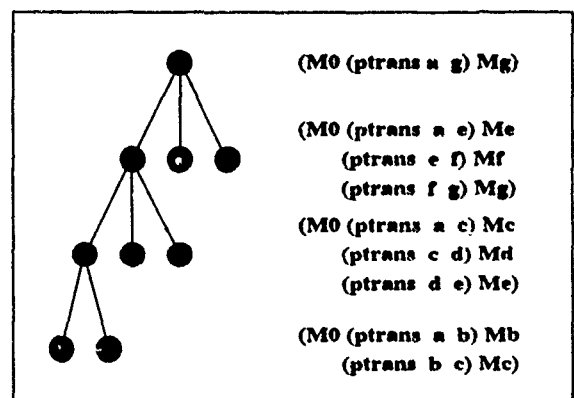


Figure 1. Goals are decomposed in a depth first fashion into less abstract subgoals, forming a tree. Each level of the tree corresponds to a plan sketch which details a subgoal in the level above it. The leaves of the tree correspond to a plan sketch of the complete task, one which is quite detailed at the beginning but becomes quite abstract towards the end.

Planning in this system is reactive. Harvey does not generate detailed plans. Plans are "sketched" and modified in response to what is perceived as a result of each action. To begin with each task given to the robot is decomposed depth first into a tree of less abstract subgoals as shown in Figure 1. The leaves of this tree form a sequence of subgoals which accomplish the task. This sequence is called a plan sketch because it is only partially developed and generally will change as execution proceeds. This plan sketch is detailed at its beginning and becomes increasingly abstract towards its end. Figure 2 shows pictorially what a tree such as in Figure 1 corresponds to for a typical goal Harvey may encounter. Action begins as soon as the first subgoal in the plan sketch is a primitive. All action is directed by the dynamic planning and execution monitor, "plan-and-monitor" [FEN 88, FEN 89] which orchestrates planning, action and perception in a fine grained, interwoven architecture.

¹This research has been supported by the Defense Advanced Research Projects Agency under RAD contract F30602-87-C-0140 and Army ETL contract DACA76-89-C-0017, and by the National Science Foundation under grant DCR 3500332.

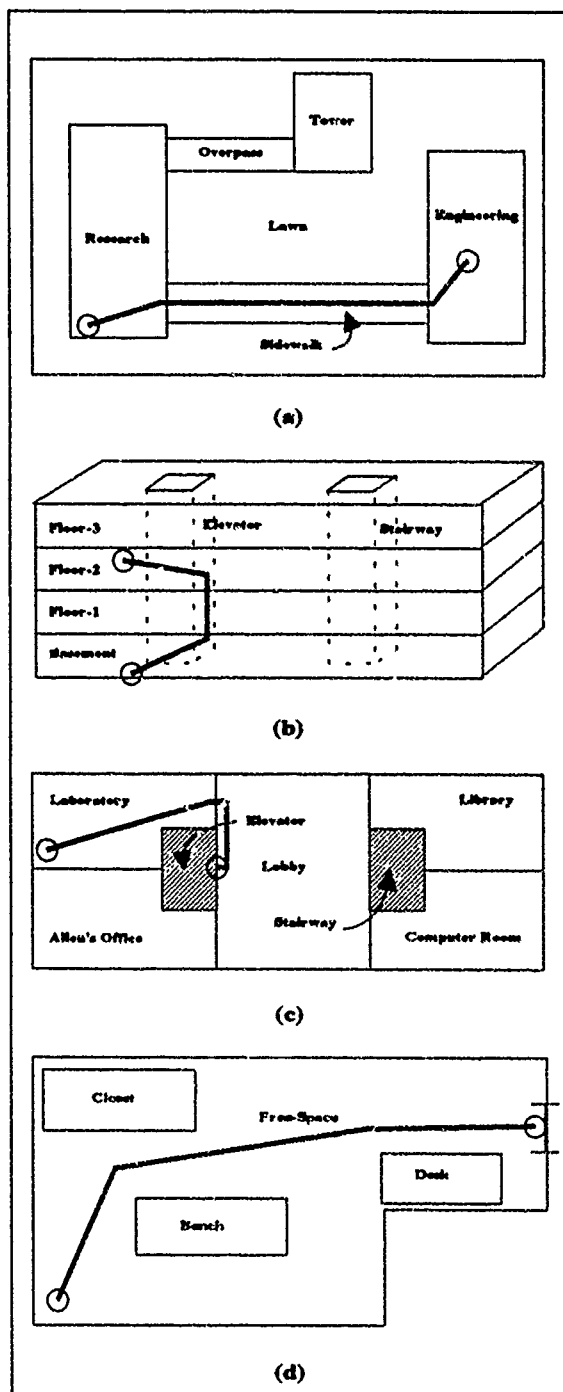


Figure 2. Accomplishing the goal (pttrans robot-lab teds-office) would begin by building a tree such as is shown in figure 1. Here the decomposition is shown pictorially from the most abstract plan sketch (a) to the most specific (d).

The plan-and-monitor executive directs planning, perception, and execution in such a way as to dynamically modify and refine the plan-sketch hierarchy to fit the actual results of each action and the details of the perceived environment. The principal activities involved in this process are: creating plan sketches, determining

milestones, milestone recognition, determination of location, and execution of primitive motor actions. Interweaving of perception, planning and action makes specific what task is expected of perception, and provides a means for focusing the knowledge available for that purpose. The result is a distribution of perception and perceptual reasoning into all aspects of navigation. Route planning uses perceptual reasoning to select appropriate perceptual milestones; plan progress is measured using perception; perception is used to relocate the robot when a milestone is not recognized; and during the execution of primitive actions, low-level perceptual feedback is used to keep the robot on the expected trajectory. The different levels of control all use model-directed vision and compare what is sensed to what is expected, issuing corrective commands to minimize any difference.

The navigation system described here differs from existing systems [THO 86, LOW 85, DIC 88a,b, BRO 86, TOS 86, HER 88a,b] in its reliance on an environmental model, high level goals, the use of sensor data in a 'verification' mode, and the close coupling of planning, action, and perception. Many of the techniques utilized by these systems (for road following, for example) could be embedded in Harvey's perceptual servoing mechanisms. Since Harvey assumes there are no unmodelled obstacles in its environment, the obstacle detection and avoidance methods developed for these systems could also be used.

3. The System Model

The system model contains environmental information relevant to the navigation task, including models of the system sensors and actuators, the environment, and the navigation task itself (the plan). The first two components of the system model are discussed briefly below; the task model is described in more detail in Section 4.

3.1 The actuator and sensor model

The actuator and sensor model contains information about the primitive actions the robot is capable of executing directly and parameters of the sensor. The suite of primitive motor actions includes only two types of intended motion: straight line forward motion and rotation about the robot's axis. All motion is ultimately decomposed into a series of such motions. This model is discussed in more detail in Section 4. The sensor model is more complex and is an ongoing research topic. Currently the model consists of: a set of camera parameters including focal length, aspect ratio, image center, and image size; a set mathematical transforms including projection and clipping; and a set of low-level vision modules capable of extracting straight lines, performing simple correlation, and the like.

3.2 The environmental model

The environmental model specifies the geometric and

perceptual structure of the robot's world. Portions of the interior of the University of Massachusetts Graduate Research Center, as well as a portion of the surrounding campus, were modelled as volumes using planar surface patches, and embedded in the locale hierarchy described below. Figure 3 is a projection of a portion of a hallway used in the experiments described in Section 5.

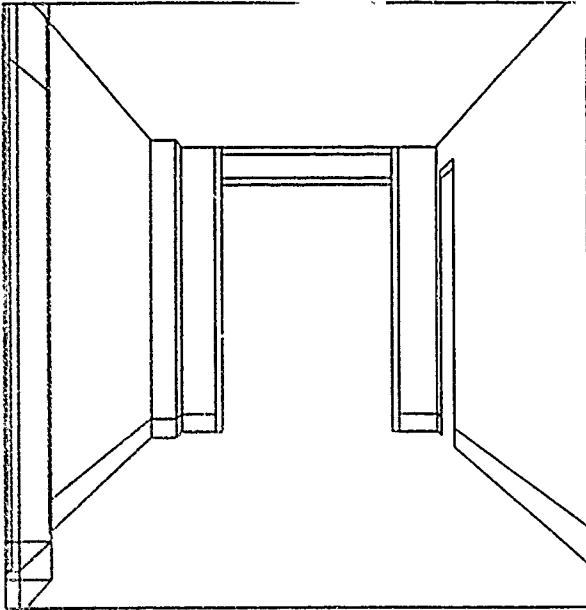


Figure 3. The model of the environment is incomplete, but positionally accurate. All obstacles are modeled but many visual details may be absent.

The environment is represented as a graph structure which captures the key topological, geometric, and physical properties of the spatial entities making up the robot's world [FEN 90]. This network describes space in terms of a hierarchical collection of "locales": spatial entities representing objects, buildings, parking lots, free space, etc. A locale is a parcel of space which has semantic significance to the navigation problem. Each locale is represented as a node in the network; arcs relate locales by means of part/subpart relations as well as by the allowable transitions between the locales (Figure 4).

The geometric properties of each face are represented in terms of lines and points, which are similarly represented. Global physical properties of a face, such as area, are kept on its property list. Visual properties, which may vary over the surface of the face, are described in terms of regions (Figure 6). Faces may have two sets of regions, corresponding to the two sides of the face. The two sets of regions account for the possible differences in appearance between two sides of a wall, for example.

4. Representing Goals and Actions

The task model represents the navigational goal(s) and how progress toward this goal is to be measured. The basic

elements of this representation are goals and milestones. Goals are represented using conceptual dependency notation [SCH 76]. The goal to move from one location to another is represented as: (ptans location-1 location-2)

Milestones are typically specifications of one or more 3D landmarks and their expected location with respect to the robot at the completion of the preceding phase of the plan sketch. Goals and milestones are collected into plan sketches represented as a list of the form:

(M0 (ptans a b) M1 (ptans b c) .. (ptans f g) Mg)

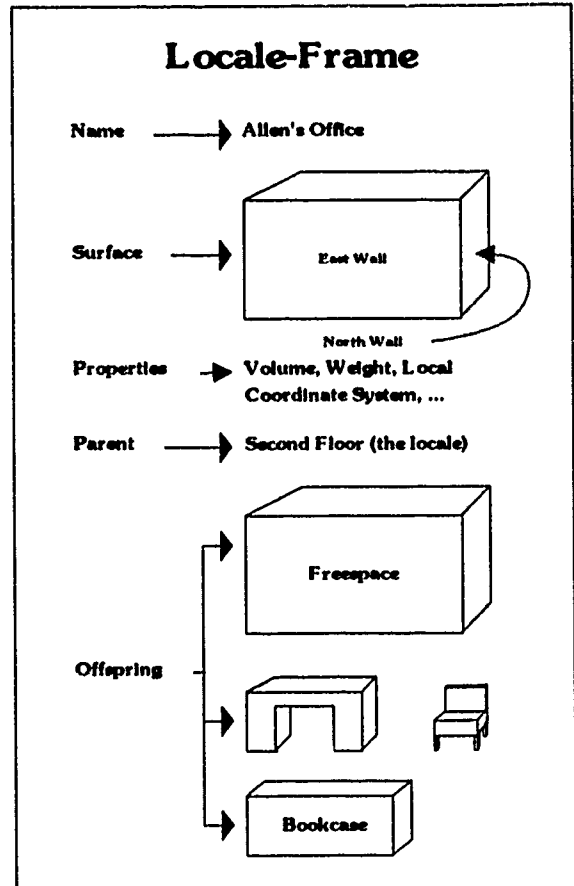


Figure 4. All 3D space entities are represented as locales. These locales are implemented as frames whose slots organize its topological, geometric, and physical properties. Shape and spatial extent are represented by a description of the locale's surfaces in terms of faces (Figure 5).

M0 is a precondition milestone which must be satisfied before the plan sketch can be considered meaningful. As motor actions take place, milestones are verified (usually visually) before the next goal of the plan can be undertaken. This ensures that the robot's actual location in the world agrees with its intended location with respect to its world model; this in turn ensures that the next goal in the plan-sketch hierarchy is applicable.

As the plan sketch hierarchy is developed, goals are refined in a depth first fashion to more and more detailed levels until the leading subgoal is a primitive subgoal. A primitive subgoal by definition is one which can be directly executed without further subgoal refinement. For the Denning Robot, the primitive actions are TURN(angle) and MOVE(distance); all primitive subgoals are decomposed into a TURN followed by a MOVE. For example, if the primitive subgoal were

(ptrans location-1 location-2)

and the path from location-1 to location-2 were unobstructed, then this subgoal could ideally be satisfied by executing the script:

```
(script-pttrans-clearpath (location-1 location-2)
  (turn (- (angle location-1 location-2)) heading)
  (move (distance location-1 location-2)))
```

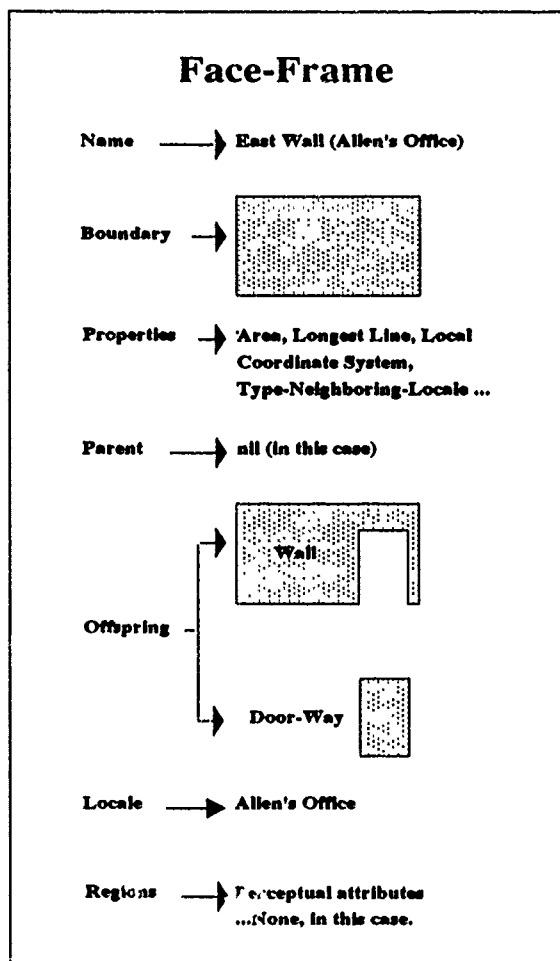


Figure 5. Locale surfaces are currently described in terms of planar faces which, like locales, are represented as frames. Values in the slots organize information such as its type (door-way or barrier) and, if it is a door-way, the name of the locale which shares the doorway.

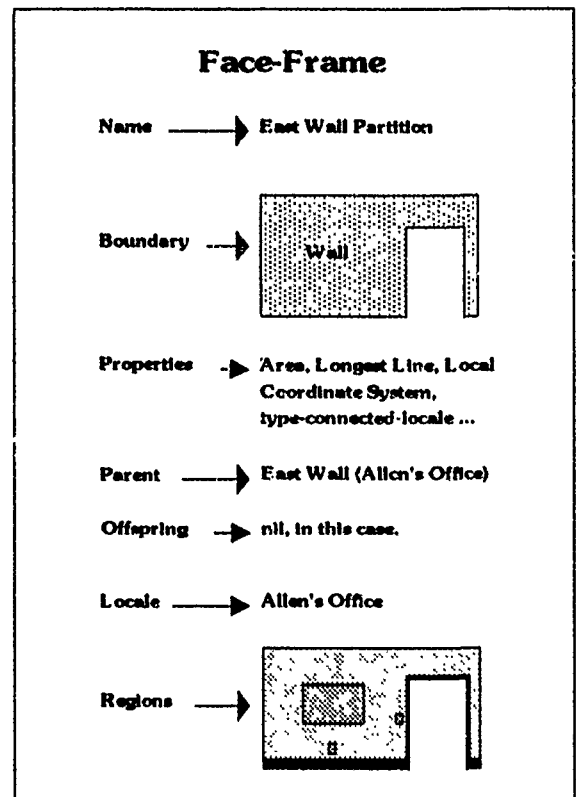


Figure 6. The appearance of each face is described as a collection of regions, each representing an area of the face which has uniform appearance properties (color, texture, etc.)

This script specifies that to get from location-1 to location-2 it is only necessary to perform two actions: first turn the difference between the angle of the line from location-1 to location-2 and the current robot heading, then move the distance between these two locations.

5.0 Perceptual Servoing

The script illustrated in section 4 will not satisfy the subgoal unless the individual actions "turn" and "move" are properly executed. In practice, this is unlikely without sensory feedback because the realities of the environment and the actuators of the agent make the execution of open loop actions unreliable. As the robot performs "turn", for example, its rubber tires introduce a translational motion or "skittering". Whenever the robot moves forward along an environmental surface, a slippery spot, a unevenness of the surface, or even a bulge in its tire may throw it off course, causing inaccurate execution. To reduce this error the execution of each action is controlled by servoing on prominent visual features in the environment. This servoing using perceptual features is called "action-level perceptual servoing".

Action-level perceptual servoing increases the accuracy of each action, but does not relate the result to progress

toward the goal, nor does it prevent the accumulation of inaccuracies over a number of such actions. It may be determined, for example, that three steps of distance 1.23 miles are required to accomplish a subgoal. Action level perceptual servoing might execute these actions fairly accurately to, say, 1.2 miles (a 2.4% error). But, after executing these three actions the agent will be .1 mile short of its goal and will very likely not even be able to see the goal from that position, so the subgoal has not been achieved. It is for this reason that milestone were introduced. A better script might be:

```
(script-pttrans-clearpath (location-1 location-2)
  (turn (- (angle location-1 location-2)) heading)
  (if (not (recognize-milestone milestone-t))
    (return (list failure last-known-location)))
  (move (distance location-1 location-2))
  (if (not (recognize-milestone milestone-t))
    (return (list failure last-known-location)))
  (return (success location-2))))
```

In this script each action is followed by a milestone check. As long as the results of these checks are positive, processing the script continues, otherwise failure is reported. In any case the last known location is returned along with the success/failure report. If the result of processing the script is (success location-2) the subgoal has been achieved. If, however, the result is (failure last-known-location) then the plan sketch hierarchy must be adjusted or redeveloped. If the location last-known-location is sufficiently near location-2, it may be enough to replace the next subgoal to be satisfied, namely (pttrans location-2 location-3), with the new subgoal (pttrans last-known-location location-3) and continue. This procedure is called "plan-level perceptual servoing".

When the distance between the last known location and location-3 is large, however, it will be necessary for the agent to relocate itself and redevelop a major portion of the plan sketch hierarchy. This relocation-redevelopment cycle forms a third level of servoing called "goal-level perceptual servoing".

Navigational tasks are thus accomplished using three nested levels of perceptual servoing: action-level, plan-level and goal-level. The details of goal-level perceptual servoing are outside the scope of this paper. The following discussion describes plan-level and action-level servoing.

5.1 Action-Level and Plan-Level Servoing

Both action-level and plan-level servoing are based on knowledge directed, top down vision. In this project it is assumed the agent has partial knowledge of all obstacles in the environment. In particular it is assumed that the agent has accurate, but not necessarily complete knowledge

about the appearance the objects in its environment. A perceptual servoing cycle begins by analyzing what is known about the environment and what should be perceived from the current location of the agent. 3D entities, called landmarks, are selected from the model on the basis of how distinctive they are, and what kind of information they offer the servoing procedure. Once these landmarks are selected their appearance is projected onto the image plane and are matched to data in the image. These matches, along with the knowledge of the 3D locations of the landmarks, are used to compute and make the appropriate corrections to the action or to the plan sketch hierarchy. Both action-level and plan-level servoing use the same landmark selection and matching procedure; they differ only in what they do with the resulting information.

5.2 Selecting Landmarks

In principle a landmark can be any 3D entity: an object, a group of objects, a group of lines [FEN 90, BEV 89, BEV 90], or cluster of surface patches, as described here. Surface patches were chosen because their reflectance patterns can be quite distinctive, making it likely that they can be isolated from other 3D entities in an image with rather simple means, such as correlation. Correlation, properly used, is a strong method for matching such reflectance patterns; it is known to be noise tolerant, and it can be computed quickly on special purpose machinery [DIC 88a,b]. Landmarks are selected on the basis that they will be easy to find using correlation.

Distinctive reflectance patterns occur where regions of differing reflectances meet, namely at boundary segments and vertices. In the following description surface patches defined by vertices are used, but the same principles apply to surface patches defined by line segments or curves. Selecting landmarks from the model, then, corresponds to searching the locale structure for vertices (equivalently lines or curves) which are surrounded by distinctive patterns. The structure of the locale network makes this computation relatively straightforward and efficient. The location of the agent is expressed in terms of a locale and its coordinate system

agent-location = (locale pose)

so it is known what locale the agent is inside. This locale, together with its offspring (freespace and the objects contained in the locale), define the scope of what can be seen by the agent: the agent can see the inside of the locale and the outside of the offspring. The search for landmarks consists of collecting the vertices (and/or lines) which make up these surfaces and selecting those which are surrounded by distinctive patterns. The surface patches used as landmarks are the vertices (lines) together with a some surrounding area.

More precisely, the procedure for selecting landmarks from the model is as follows:

1. From the locale corresponding to the agent's current location collect all the vertices associated with the regions on the inside surface of the locale and the outside surface of each offspring locale. This is accomplished by looking at the surface description for each locale:

locale --> surface --> faces --> regions --> lines --> vertices

2. Delete vertices which are not expected to be visible to the agent from its current location. This is done by first clipping the projection to the image plane (ignoring occlusion) and then deleting occluded vertices from what remains.

3. Discard vertices which do not touch two regions which differ in reflectivity by some threshold. These reflectivities can be found by following pointers to the associated regions:

vertices --> lines --> regions --> reflectivity

4. Return the remaining vertices.

Landmarks correspond to the vertices resulting from this procedure. These are the landmarks used in action-level and plan-level perceptual servoing. In both types of servoing, knowledge of the reflectances of these landmarks is used to construct the appearance templates used to identify the landmarks in the image data.

5.3 Constructing Appearance Templates

Appearance templates are image arrays which specify how a landmark should appear in the image. For the vertex landmarks described in the previous section, these are $n \times n$ image arrays centered on the image of the vertex with pixel values determined by the geometry, reflectance values and the agent's location as represented in the model. Constructing these arrays is a localized rendering process.

Templates are constructed by ray tracing, considering only those surfaces and regions whose boundaries pass through the vertex. The pixel values in the template array are assigned to be proportional to the reflectance value of that region which the ray strikes first. A more precise statement of this procedure is:

1. Collect the regions which form the vertex. This is accomplished by following pointers in the locale network, beginning with the vertex frame:

vertex --> lines --> regions

2. For each pixel in the appearance template there is a ray which starts at the focal point of the camera lens and passes through the center of the pixel. Find the 3D intersection of this ray with each of the regions

collected in (1).

If the ray does not intersect any of the regions either the patterns associated with the landmark are too detailed and may be subject to aliasing or the vertex is on an occluding edge of an object and certain pixels will be unpredictable. The use of the landmark will be error prone so it should not be used. In this case a null template is returned.

Otherwise assign to the pixel the value $255 \cdot R$, where R is the reflectance of the region whose point of intersection with the ray is closest to the camera lens focal point.

3. The resulting array is the appearance template.

The resulting template is used match the landmarks with their projections in an image. This matching is done using correlation in a way which matches reflectance values, rather than intensities, to remove the ever present effects of uneven illumination.

5.4 Matching Templates to the Data

Correlation is a well understood mathematical tool which has been widely used in signal processing and in 2D image processing, but it has not been used as much in 3D scene processing because there are several severe problems which arise. It is easy to describe these problems if we think of an image as a function $f(x,y)$ on the x - y plane. Correlation is a measure of how similar two such functions are. Images of 3D scenes are, however, strongly effected by several factors, all significant in the kinds of scenes our agent will encounter.

1. The shape of the image of an object varies as the viewpoint is changed, due to projection effects.
2. Changes in viewpoint alter what can be seen in the image of a scene, due to occlusion effects.
3. Changes in lighting modify the intensity of the image, either locally or globally.
4. Specularities vary, sometimes strongly, with lighting and viewpoint

All four problems affect the nature of the image function $f(x,y)$ corresponding to a scene in such a way that its "shape" and "height" may vary considerably. This makes correlation useless for global scene matching and makes local scene matching difficult, at best. Knowledge of the agent's approximate location, together with knowledge of how these problems affect the image function makes it possible to cope with these problems. Problems 1 and 2 are managed by the way the appearance templates are constructed. The perspective distortions of 1. and the occlusion effects of 2. are kept to a minimum by the ray tracing procedure and the rejection test in step 2 of the

construction method described above.

The third problem is managed by correlating reflectance values, rather than intensity values. For sufficiently small patches in the environment it is assumed that the light intensity is constant. This is not an unreasonable assumption for an environment which is not highly textured by shadows. The values of the pixels $P(i,j)$ corresponding to such a patch can be expressed as:

$$P(i,j) = I * R(i,j)$$

where I is the (constant) light intensity over the surface patch and $R(i,j)$ is the average reflectance over the surface which contributes to the pixel value $P(i,j)$. Under these circumstances the effects of the illumination can be removed by dividing each pixel in the image by the sum of the pixel values over the patch. The new array $RP(i,j)$

$$\begin{aligned} RP(i,j) &= \frac{P(i,j)}{\sum_{k,l=0}^n P(i,j)} = \frac{I * R(i,j)}{\sum_{k,l=0}^n I * R(i,j)} \\ &= \frac{I * R(i,j)}{I * \sum_{k,l=0}^n R(i,j)} = \frac{R(i,j)}{\sum_{k,l=0}^n R(i,j)} \end{aligned}$$

consists of values which are independent of the illumination incident on the surface and depends only on the reflectance properties of the surface itself.

Management of the fourth problem, specularities, is still a matter for future research. Currently it is assumed that the number of landmarks unrecognized due to specularities will be minimal. This assumption seems to be well justified in our experiments to date.

Matching itself is done using normalized correlation. Normalized image patches $RP(i,j)$ are matched against normalized appearance templates $RT(i,j)$ using what is known as normalized correlation:

$$NC(i,j) = \frac{2 * \sum_{k,l=0}^n RP(k-i,l-j) * RT(k-i,j-l)}{\sum_{k,l=0}^n RP(k-i,l-j)^2 + \sum_{k,l=0}^n RT(k-i,j-l)^2}$$

The implemented computation differs somewhat from this for efficiency reasons, but only in the order in which the elements are computed. This correlation method has proven to be very reliable for locating landmark images in the indoor environments used for the experiments described below. Outdoor experiments are planned for the near future.

Once the selected landmarks have been located in the image we have matched the image data to the 3D landmark models. Next appropriate corrective actions are taken. These actions are different for action-level and plan-level

serving, so we describe them separately.

5.5 Action-Level Perceptual Servoing

Actions are carried out incrementally, using the location of landmark images to compute necessary corrections. Each increment begins by selecting landmarks and matching their projections with data in the image. By measuring the discrepancy between where the features should be and where they are determined to be in the image, it is possible to compute the corrective action required to bring the positions into agreement (Figure 7). This perceptual servoing has the effect of locking the robot onto a trajectory which improves the accuracy of the actions over that which would be obtained without servoing.

The "move" action illustrates the action-level perceptual servoing principle well. Experiments have shown that during an open loop execution of an intended straight line forward motion the robot vehicle moves in a curved path. Sufficiently small incremental paths can be approximated by a straight line at some angle with respect to the intended line. Figure 8 depicts the geometry of the situation.

If we let

- m = incremental distance moved
- e = distance "off desired line" after an incremental move of distance m
- q = shortfall in distance covered along intended line.
- x = x coordinate of the landmark image (in camera coordinates)
- f = focal length of camera lens
- X = x-coordinate of landmark in robot coordinates (expected)
- Y = y-coordinate of landmark in robot coordinates (expected)
- h = heading error after the incremental move.
- b = measured landmark bearing

Then

$$h + b = \tan^{-1} \left(\frac{Y-e}{X+q} \right)$$

so

$$h = \tan^{-1} \left(\frac{Y-e}{X+q} \right) + \tan^{-1} \left(\frac{-x}{f} \right)$$

Thus from the observed x-coordinate of the landmark image we can estimate the heading error, provided the values of e and q do not distort the value of

$$\left(\frac{Y-e}{X+q} \right).$$

This is typically the case for small incremental motions. In our configuration e is on the order of .02 ft and q is on the order of .05 ft. Hence for landmarks more distant than 3 ft this ratio can be well

approximated by (X/Y)

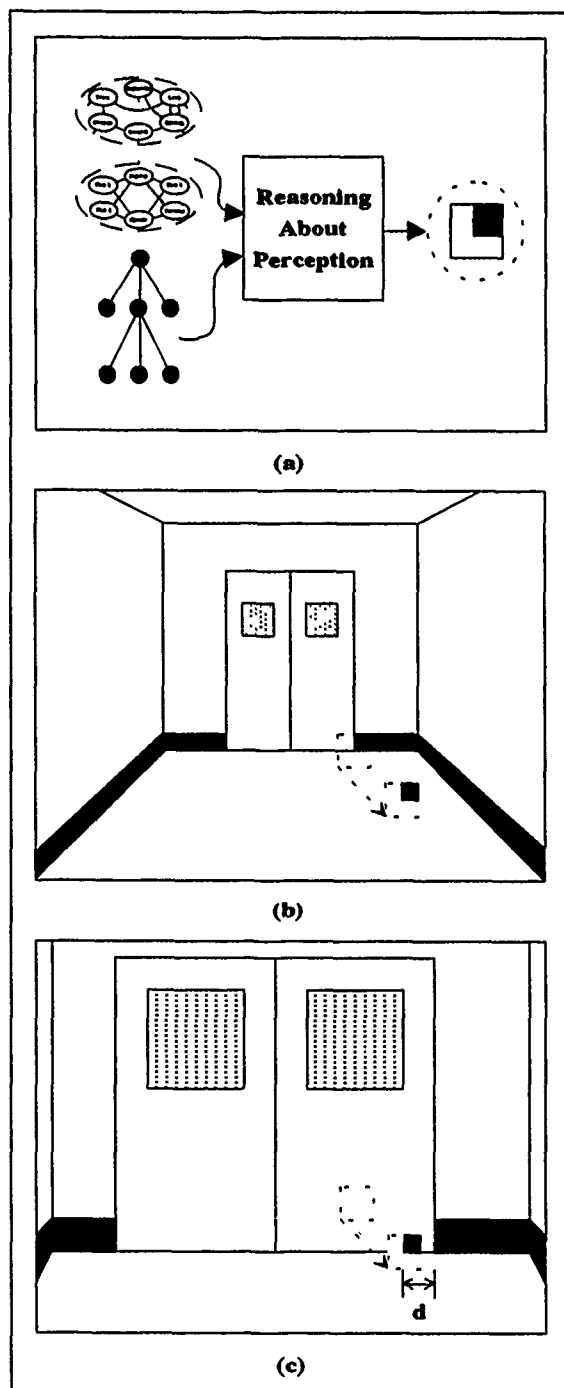


Figure 7. Before each action begins, landmarks are selected (a) using knowledge of the environment and the task. An incremental action is taken and the motion of the landmark is predicted (b). Any difference between this projection and the actual location of the landmark image (c) is used to determine what corrections to make.

Given an approximation to the heading error the corrective action is determined according to the geometry indicated in

Figure 9. The quantities e and q are given by the equations:

$$e = m \cdot \sin(h)$$

$$q = e \cdot \tan(h)$$

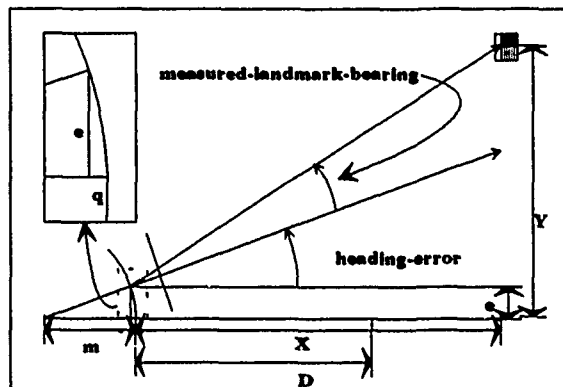


Figure 8 Experiments have verified that the motion of the robot can be approximated to be a straight line for very small distances. Under these assumptions the geometry of an incorrect motion allows us to compute the heading error from the position of a landmark in the image.

If the agent motion is corrected by steering to a point on the intended line a distance D away from the expected location we can correct as follows:

$$\text{correction} = - \left[h + \tan^{-1} \left(\frac{e}{D+q} \right) \right]$$

$$= - \left[h + \tan^{-1} \left(\frac{m \cdot \sin(h)}{D + m \cdot \sin(h) \cdot \tan(h)} \right) \right]$$

The results of a simulation using this method are shown in Figure 10, both for a camera which is perfectly aligned with the robot motion, as was assumed in the above description, and for a misaligned camera. Aligning the camera to the robot motion can be a tricky operation, but it is only a modest effort to align within 0.005 radians, so the simulation predicts fairly accurate control.

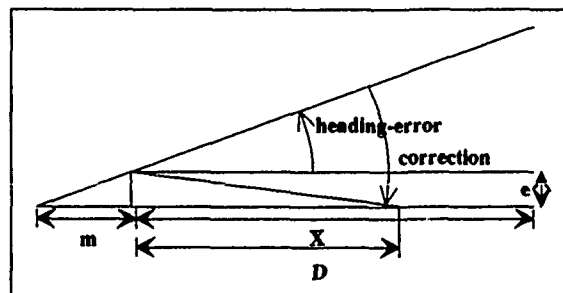


Figure 9 Once the heading error has been determined, the robot is turned toward a distant point on its original intended path.

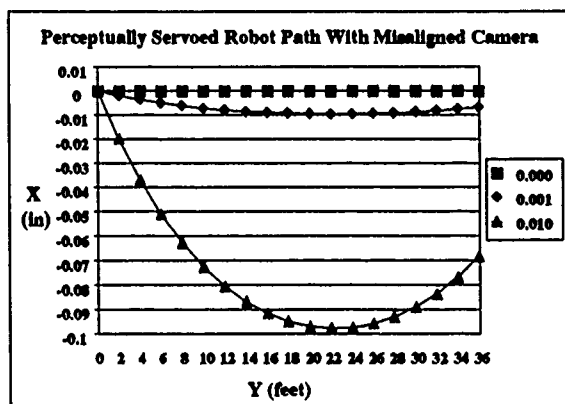


Figure 10. A simulation of the action-level perceptual servoing concept shows the effects of camera misalignment on motion control for 0.000, 0.001 and 0.010 radians of misalignment.

To verify this prediction several experiments, both with and without action-level perceptual servoing, have been run. In the experiments Harvey was to roll along a straight line 40 feet long, marked on the floor of a Graduate Research Center hallway. The vehicle was stopped every two feet and its deviation from the marked line was measured. This experiment was run a number of times; the results shown in the graph of Figure 11 represent the best unservoed result compared with a typical servoed result. Even after a rather painstaking setup procedure the unservoed vehicle wandered over two inches from the line within the first 20 feet. Other runs resulted in as much as a foot deviation in unservoed mode. Unservoed trials were stopped at around 20 feet because the vehicle was significantly off course and the total deviation was increasing. In contrast, in servoing mode the vehicle stayed within .3 inch of the line for the full 40 feet.

These experiments have only been performed indoors, but the results of these experiments are encouraging and support the use of perceptual servoing to control motion over reasonable navigation segments. It seems possible that each motion can be carried out accurately enough to support the assumptions made in the next section.

5.6 Plan-Level-Perceptual Servoing

Plan-level perceptual servoing uses the same landmark selection and matching procedures as action level perceptual servoing. Consequently, since the agent has been tracking the landmarks for steering purposes, it is likely that the matching effort during milestone recognition will be small, since the images of the landmarks at milestone recognition time will be known. The resulting matches and the 3D model information are used to determine the robot's location using a 3D pose refinement algorithm [KUM 89]. If the robot location as determined by the pose refinement agrees with what is expected then

the milestone has been satisfied and there is no need to modify any subgoals.

If pose refinement and expectations differ by a small amount, then the first location in the next subgoal is adjusted to reflect this difference. Currently the next subgoal is changed from (ptans location-1 location-2) to (ptans pose-determined-location location-2). Plan-and-monitor automatically takes care of any detailed plan refinement if this change makes it necessary. Should the pose-determined location differ greatly from expectations, or if it should fail to determine a location, control is passed on to goal-level perceptual servoing.

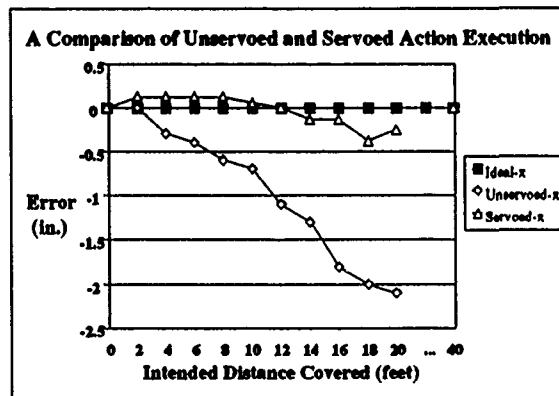


Figure 11 Experimental results have shown the ability to keep the robot within .3 inches of its intended course by using action-level perceptual servoing. Without servoing the robot wanders two inches from the path by the time it has traveled 18 feet.

To use position information in this way, how accurate must the pose returned by the pose refinement step be? Since the robot has been servoing on image features over the previous navigation leg, it is likely that it will be fairly close to its expected position, say within 6-12 inches. Within this area of uncertainty, pose refinement should return an accurate value, say within an inch or two. Outside of this area, it is likely that significant subgoal redevelopment will be necessary, so it is appropriate for plan level servoing to pass control up to goal-level servoing. Determining the robustness of combined action-level, plan-level and goal-level servoing over a wide variety of environmental conditions is the subject of ongoing experiments.

Experimental results in support of using correlation matching and pose refinement for this purpose have been promising. Table 1 shows results from an experiment to determine the accuracy of this approach. Using a single image taken in a hallway, the expected location of the robot was varied to create the effect of being off target. The table shows the measured pose for different expected locations, corresponding to the actual location of the robot (y0) when the image was taken and three incorrect

expected locations: to the left 3 inches (y0-3), to the left 6 inches (y0-6) and to the right 3 inches (y0+6) .

	y0-6	y0-3	y0	y0+3
Expected				
x	40.00	40.00	40.00	40.00
y	3.50	3.75	4.00	4.25
z	0.00	0.00	0.00	0.00
theta-x	0.00E+00	0.00E+00	0.00E+00	0.00E+00
theta-y	0.00E+00	0.00E+00	0.00E+00	0.00E+00
theta-z	3.14E+00	3.14E+00	3.14E+00	3.14E+00
Measured				
x	39.93	39.93	40.11	39.85
y	4.01	4.01	4.08	4.17
z	-0.22	-0.22	0.20	0.00
theta-x	-4.69E-03	-4.69E-03	-9.00E-03	-4.95E-03
theta-y	-1.51E-02	-1.51E-02	-5.00E-03	-9.25E-03
theta-z	3.14E+00	3.14E+00	3.14E+00	3.14E+00

Table 1 Using correlation for landmark matching, together with 3D pose refinement experiments have shown the ability to determine the robot's location to within 1.5 inches. These results compare the expected (incorrect) location with what was measured. The ground truth location in all cases was the same $x=40$, $y=4$, $z=0$, $\theta_x=0$, $\theta_y=0$ and $\theta_z=3.1415$.

The experimental results discussed in sections 5.5 and 5.6 are quite encouraging. Each of the perceptual servoing algorithms seems to have significant potential for robustness when examined individually. However, they effectively complement each other when combined into a system. The results of a multiple leg experiment in the hallway environment is described in [FEN 90]. In this particular experiment, the vehicle stopped within 1 inch of the final goal.

6.0 Conclusions

The preliminary indoor experimental results presented here support the use of perceptual servoing to increase the accuracy of primitive action execution over open loop execution and to maintain the position of the robot relative to its internal model. As weather permits, similar outdoor experiments will be performed. The results obtained from plan-level perceptual servoing using correlation matching compare favorably with a second technique using 2D line correspondences [FEN90] but is computationally less expensive. Both approaches use the 3D pose refinement mechanism developed by Kumar [KUM 89] to compute the pose and orientation of the robot with respect to the internal model from the landmark matches. The robustness of these techniques is currently being explored.

References

- [BEV 89] J. R. Beveridge, R. Weiss and E. Riseman, "Optimization of 2-Dimensional Model Matching Under Rotation, Translation and Scale," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR 89-57, June, 1989.
- [BEV 90] J.R. Beveridge, R. Weiss, and E.M. Riseman, "Combinatorial Optimization Applied to Variable Scale 2D Model Matching," 10th International Conference on Pattern Recognition, Atlantic City, NJ, June 1990, to appear.
- [BRO 86] R. Brooks, "A Robust Layered Control System for a Mobile Robot," IEEE Journal of Robots and Automation, Vol. RA-2(1), pp. 14-23, 1986.
- [DIC 88a] E. Dickmans and V. Graft, "Applications of Dynamic Monocular Machine Vision," Machine Vision and Applications, Vol. 1, pp. 241 and following, 1988.
- [DIC 88b] E. Dickmans and V. Graft, "Dynamic Monocular Machine Vision," Machine Vision and Applications, Vol. 1, pp. 223-240, 1988.
- [FEN 88] C. Fennema, E. Riseman and A. Hanson, "Planning With Perceptual Milestones to Control Uncertainty in Robot Navigation," Proc. of SPIE -- International Society for Photographic and Industrial Engineering, Mobile Robots III, Cambridge, MA, pp 2-18, 1988.
- [FEN 89] C. Fennema, A. Hanson, E. Riseman. (1989b). "Towards Autonomous Mobile Robot Navigation," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR 89-104, October 1989. Also appeared in Proc. DARPA Image Understanding Workshop, pp219-231, May1989
- [FEN 90] C. L. Fennema, A. Hanson, E. Riseman, J. Beveridge and R. Kumar, "Towards Autonomous Navigation", IEEE Systems, Man and Cybernetics, 1990, to appear.
- [HER 88a] M. Herman and J. Albus, "Overview of the Multiple Autonomous Underwater Vehicles (MAUV) Project," Proc. IEEE International Conference on Robotics and Automation, Philadelphia, PA, pp. 618-620, April 1988.
- [HER 88b] M. Herman, T. Hong, S. Swetz, D. Oskard, and M. Rosol, "Planning and World Modeling for Autonomous Undersea Vehicles", Proc. Third IEEE International Symposium on Intelligent Control, Arlington, VA, August 1988.
- [KUM 89] R. Kumar and A. Hanson, "Robust Estimation of Camera Location and Orientation from Noisy Data having Outliers," Proc. of IEEE Workshop on Interpretation of 3D Scenes, Austin Texas, pp. 52-60, November 1989. A more detailed version may be found in Dept. of Computer and Information Sciences, University of Massachusetts (Amherst), TR89-120, December 1989.
- [LOW 85] J. Lowrie, M. Thomas, K. Gremban and M. Turk, "The Autonomous Land Vehicle (ALV) Preliminary Road-Following Demonstration," Proc. of Intelligent Robots and Computer Vision (SPIE 579), D.P. Casasent, Ed., pp. 336-350, 1985.
- [SCH 76] R. Schank and R. Abelson, "Scripts, Plans, Goals and Understanding", Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1976.
- [THO 86] C. Thorpe, S. Shafer, and A. Stentz, "An Architecture for Sensor Fusion In A Mobile Robot," Proc. of IEEE International Conference on Robotics and Automation, Vol 3, pp. 1622 and following, April 1986.
- [TOS 87] G. Toscani and O. Faugeras, "Structure from Motion Using the Reconstruction and Reprojection Technique," Proc. of IEEE Workshop on Computer Vision, pp. 345-348, 1987.

IMAGE-BASED NAVIGATION USING 360° VIEWS*

Jia-Wei Hong and Xiaonan Tan
Robotics Research Laboratory
Courant Institute, New York University

Brian Pinette, Richard Weiss and Edward M. Riseman
Computer and Information Science Department
University of Massachusetts at Amherst

Abstract

Mobile robot navigation has proved to be a difficult task, even when a robot is given a three-dimensional model of its environment. When the robot must also be capable of acquiring a model of its environment, the construction of a robot navigation system becomes even harder. Our system solves the model acquisition problem by representing the environment as a set of snapshots of the world taken at target locations. The robot navigates by using an image-based local homing algorithm to move between neighboring target locations. This paper describes an approach that divides large-scale navigation tasks into a sequence of small-scale navigation tasks that are solved by local homing. Some interesting and novel features of our approach are an imaging system that acquires a compact, 360° representation of the environment and an image-based, qualitative homing algorithm that allows the robot to navigate without explicitly inferring three-dimensional structure from the image. We describe the results of an experiment in a typical indoor environment and argue that image-based navigation is a feasible alternative to approaches using three-dimensional models.

1 Introduction

Mobile robot navigation has proved to be a complex task, even in a task domain where the robot is given a detailed three-dimensional model of its environment (Fennema et al [6]). Providing a robot with such a model is itself a significant, time-consuming task: a survey of many of the natural and cultural objects in the robot's environment and their spatial relationships to each other is required. If done sparsely this might amount to the extraction of key landmarks that would allow proper navigation relative to the prominent features of the visible environment. In the limit, this would involve determining contour maps and full three-dimensional solid models of all prominent objects. In either case, acquiring accurate geometric information is difficult and expensive.

Because it is difficult to acquire world models for navigation, it becomes an obvious goal to have a mobile robot

explore the environment itself in order to extract information sufficient for effective navigation (e.g., Davis [4] or Yeap [14]). There are many possible ways this might be done, but if the goal is navigation — instead of full three-dimensional surface reconstruction — then the environmental features that are most prominent and visible (i.e., landmarks) will provide the key information for locating the robot vehicle and determining an appropriate path to the goal. Recent efforts involving navigation with landmarks include Fennema et al [6], Kumar and Hanson [10], and Zheng and Tsuji [15].

In this paper we develop a navigation strategy built on the technique of image-based *local homing*. Homing is a navigation task in which the goal is one of a fixed set of target locations known to the robot. The robot is capable of finding its way only to these target locations, but not to any arbitrary place in its environment. In contrast, such tasks as "Go down Elm St. until you come to a big white house with a poplar tree in front" or "Move three meters north" are not homing tasks; they require the robot to move to unfamiliar locations.

We use a novel and powerful imaging system to project a full 360° view of the world into a single image and then condense this view into a compact, one-dimensional *location signature*. A location signature retains enough information about the landmarks seen from its target location to allow homing. In image-based local homing, the differences between the signature of a robot's current location and the signature of a target location are used to compute incremental movements that take the robot closer to the target location. We call our technique "local" homing because the robot's current location must be close enough to the target location that it falls within its "capture region" for homing. If the robot's current location is too far from the target location, the homing algorithm will fail because there will be too much distortion in images of the prominent landmarks common to both location signatures.

We acquire a model of the world by running the robot along a desired route and having the system extract location signatures for a sequence of target locations on the route. After acquiring this model, the robot can navigate the route by successively homing to each of its target points. Thus we have reduced the problem of navigating a large-scale space to a problem of navigating a sequence of small-scale spaces. Figure 1a shows

*This research has been supported by the Defense Advanced Research Projects Agency under RADC contract F30602-87-C-0140 and Army ETL contract DACA76-89-C-0017.

schematically how a route would be segmented so that each target location falls within the capture region of the following target location. It would also be possible to establish a two-dimensional spatial sampling of target locations. Figure 1b schematically shows the capture region for a single target location (the black dot) in a small portion of such a sampling. (Capture regions for other locations — denoted by white dots — are not shown.) Note that the capture region of the target location includes many of its neighbors. A robot could navigate between any two points in a sampled region by homing along a sequence of such neighboring target locations.

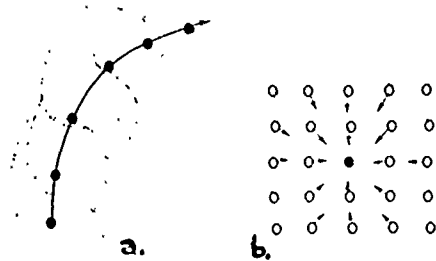


Figure 1: Segmenting the world into capture regions. a) Navigation of a route via a sequence of sampled target locations along the route. b) Navigation in general two-dimensional space by locally homing across a two-dimensional region of sampled target locations.

2 Related Research

This work is related to three general areas of research: motion analysis, associative homing, and landmark-based navigation. Image-based local homing is similar to such computer vision tasks as "relative orientation" (Horn [8]) and "structure from motion" (e.g., Ullman [13] and Adiv [1]). Like these tasks, it uses two-dimensional images to make decisions about the three-dimensional structure of the world. It differs from these tasks, however, in two ways that make the techniques we present practical, efficient and robust. First, it does not compute the exact direction or distance to the target location, but computes instead an approximately correct movement that will take the robot closer to a target location. Second, it does not build a three-dimensional model of the world (such as a depth map) and then reason with that model. Instead, it reasons directly from the images. Thus it has the advantages of being a qualitative, image-based approach rather than a quantitative, model-based approach. Qualitative approaches have the potential to be robust, since measurement noise is unlikely to change a qualitative description of the world. Image-based approaches are fast, since they avoid the extra time spent in computing three-dimensional models from images and merging them into a three-dimensional

description of the world (e.g., Ayache and Faugeras [2]).

Nelson [11] and Zipser [16] have explored qualitative, image-based approaches to *associative homing*. In *associative homing*, snapshots are taken at many reference locations in the environment. Stored with each snapshot is the movement vector that will take the robot from that reference location towards the target location. Nelson's system compares the current image to all stored snapshots and has the robot make the movement associated with snapshot that best matches its current view. Zipser's system also compares the robot's current store to all stored snapshots. Instead of picking the best match, however, it averages all movements vectors, weighting each by the degree-of-match of its snapshot. The difference between associative homing and local homing is that associative homing algorithms work by retrieving stored homing vectors, but local homing algorithms work by computing homing vectors on the fly.

Zheng and Tsuji [15] are exploring an approach to landmark-based navigation that is similar in many ways to ours. It uses a rotating slit scanner to produce a 360° panoramic view. Like us, they store a sequence of images along a path and use a relatively simple feature-matching strategy. Unlike us, they do not compress their images, so the amount of storage they need to store a path can grow quickly with the length.

Others, while using three-dimensional models to do landmark-based navigation, are investigating ways to avoid some of the costs of model-based techniques. Dickmanns and Graefe [5] use Kalman filtering of the image to directly update changes to a three-dimensional model of the world rather than computing the inverse perspective transformation; they also track image features to avoid computing the forward perspective transformation. Fennema et al [6] use three-dimensional models to generate the image the robot would see at the target location; the robot then servo's directly on the image features, tracking them via correlation. This frees the robot from doing pose refinement (i.e., updating the location and orientation of the robot) at every step. Breuel [3] uses image-derived features rather than partial three-dimensional descriptions of objects to index into a three-dimensional model base.

The remainder of this paper will provide the details of matching landmark features and computing local homing movements for navigation. We will also present results of indoor experiments that acquire environmental information in a training phase and then use this information for successful navigation.

3 Acquiring and Processing Images

3.1 The Robot and its Imaging System

We have implemented the homing algorithm on a Denning DRV-1 mobile robot. Although the three wheels of the DRV-1 can be steered, the body of the robot maintains a fixed orientation. Thus the robot cannot intentionally change the orientation of its body although this orientation may be accidentally changed (e.g., when the robot travels over rough ground). In our applica-

tion the imaging system is attached to the robot's body. The homing algorithm takes advantage of this fact and assumes that the robot's perceptual frame of reference rotates very little as the robot moves through the world.

The imaging system (Figure 2), which is mounted to the front of the robot's body, comprises a spherical mirror mounted above a video camera. The video camera points up at the bottom of the spherical mirror and sees a 360° "hemispherical" image of the world (e.g., Figure 3). This imaging technique is similar in principle to the conical mirror and laser striping system of Jarvis and Byrne [9], but different from other methods that rotate a horizontal camera to acquire a panoramic view (e.g., Zheng and Tsuji [15] and Suzuki and Arimoto [12]).

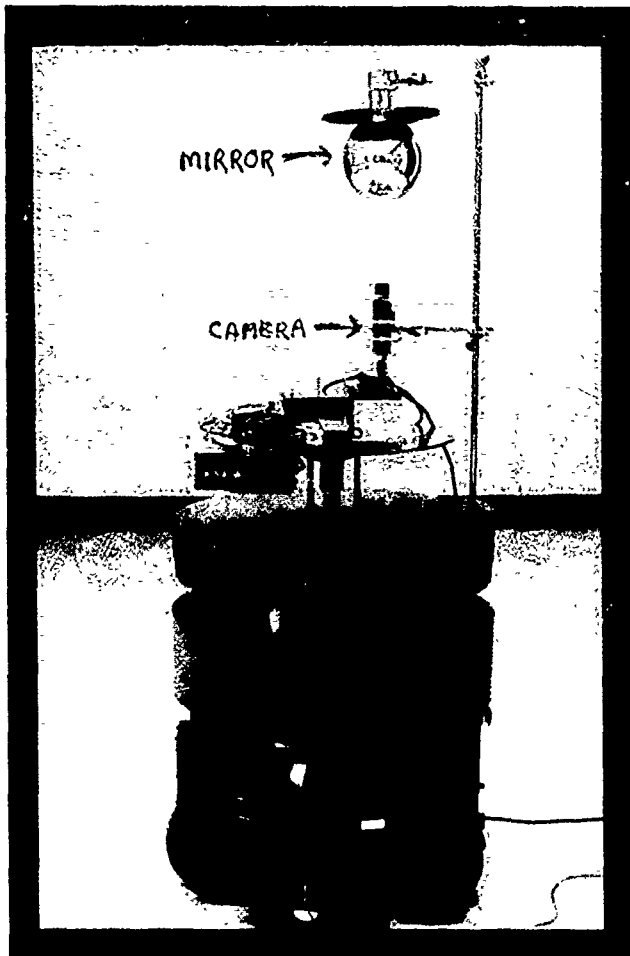


Figure 2: Robot with imaging system.

3.2 Extracting Location Signatures from the Hemispherical Image

The spherical imaging system so greatly distorts the scene during projection that the image changes dramatically as the robot moves. There is, however, a projective invariant on the horizon line. The robot's horizon lies in a circle in the hemispherical image shown by the annulus of tick marks in Figure 3. As the robot moves horizontally across the ground plane, landmarks that project to points inside the circle can potentially move to any loca-

tion inside the circle; landmarks that project to points outside the circle can potentially move to any location outside the circle. Landmarks that project to points on the horizon circle, however, remain on this circle as the robot moves. By sampling the image on the horizon line, we not only take advantage of a geometric invariance, but we reduce the dimensionality of the landmark matching problem.

The robot extracts a one-dimensional, circular location signature by sampling the hemispherical image along the horizon circle at angular intervals $\Delta\theta$; in our experiments, we sampled at 1° intervals. Each sample is a radial average of the image near the horizon circle; in our experiments, we average over 5 radial pixels. We can formally express the relation between the one-dimensional location signature V and the two-dimensional image I as

$$V_i = \sum_{j=-2}^2 I(i\Delta\theta, r_h + j\Delta r),$$

where V_i is the i^{th} intensity value of a one-dimensional location signature V , $I(\theta, r)$ is the intensity of the hemispherical image at polar coordinates (θ, r) , $\Delta\theta$ is the angular sampling interval, Δr is the radial sampling interval and r_h is the radius of the horizon circle. Thus we compress a 512 by 512-byte image into a 360-byte location signature. This efficient representation of images is of major importance in the simple yet effective development of the homing algorithm presented in later sections.

Figure 3 shows the annulus of pixels our robot samples in the hemispherical image to get a one-dimensional location signature. The short tick marks show location of the actual samples. Note that we have extended every tenth tick mark and added cross-hairs. These serve as visual aids and do not represent locations of image samples. Figure 4 shows a graph of a typical location signature. The graph represents the intensity profile along the horizon circle in Figure 10; the circle is sampled in a counter-clockwise direction starting from the left side of the horizontal axis.

3.3 Finding Characteristic Points in the Location Signature

Let us now examine how prominent world features, i.e., landmarks, are selected from the location signature. We call these features *characteristic points*. In the remainder of the paper we will sometimes use the terms "characteristic points" and "projections of landmarks" interchangeably. Characteristic points of V are found in three steps. On the first step the location signature gets segmented into regions of monotonically increasing or decreasing intensity. On the second step the point of maximum instantaneous intensity change in each segment is found. Such a point is accepted as a potential characteristic point if it represents a large enough instantaneous intensity change or if the total intensity change across its segment is large enough. On the third step these potential characteristic points are ranked, and the top fifteen are selected as those image features that represent

as $w_\ell = 1/|2 + \ell|$, that gives more weight to the center of the window than to the edges. In our system, the window into the location signature is 13 pixels wide. We say that σ is an approximation to the standard deviation because it computes it with the L_1 norm (i.e., absolute value) instead of the L_2 norm (i.e., squared value). Note that this transformation compensates only for variations in brightness; it does not compensate for spherical distortions induced in the images of landmarks as the robot moves.

We match the characteristic points in the current signature V against the image points (not necessarily characteristic points) in the target signature V^T . The actual matching function we use is

$$\rho_{i,j} = \sum_{\ell} w_{\ell} |(V_{i+\ell} - \mu) - \frac{\sigma}{\sigma^T} (V_{j+\ell}^T - \mu^T)|,$$

where $\rho_{i,j}$ is the match value at current location signature index i and target location signature index j .

5 Determining the Homing Movement

The difference between the bearing of a characteristic point in the signature for the robot's current location and bearing of the point it matches in the target signature is the *offset* ϕ of the characteristic point. The offsets for the characteristic points allow the algorithm to compute an incremental local homing movement. Suppose the robot is at some current position O and its goal is target position M , as shown in Figure 6. What would the robot see if it were to move directly towards target location M ? The spherical mirror induces very simple image displacements of landmarks that project onto the horizon line. Every characteristic point on the invariant horizon circle slides along the horizon circle away from the robot's direction of motion, as shown in Figure 7. Thus the robot should move in a direction that will cause the characteristic points to slide to the bearing they have at the target location.

Let us consider landmark A in Figure 8. The offset between A in the current location signature and its matching point A^T in the target location signature is ϕ_A . If the robot were at the target location, this offset would be 0. Our strategy is to move the robot in a way that most quickly decreases this offset — a direction ω_A perpendicular to the bearing θ_A of characteristic point A in the current location signature (Figure 8). By similar reasoning, the fastest way to reduce the offsets ϕ_B and ϕ_C of characteristic points B and C is to move in directions ω_B and ω_C , respectively. Figure 8 shows how the homing vectors ω_B and ω_C (shown as thin dotted lines) add to the homing vector ω_A (shown as a thick solid line) to form the final homing vector ω (shown as a thick dotted line). The final direction vector ω , a sum of individual homing vectors, points in the approximate direction of the target location T ; the true direction towards T is ω' (shown as a thick dotted line).

This final direction is computed by

$$\omega = \arctan \left(\frac{\sum_i -\text{sgn}(\phi_i) \sin(\pi/2 + \theta_i)}{\sum_i -\text{sgn}(\phi_i) \cos(\pi/2 + \theta_i)} \right),$$

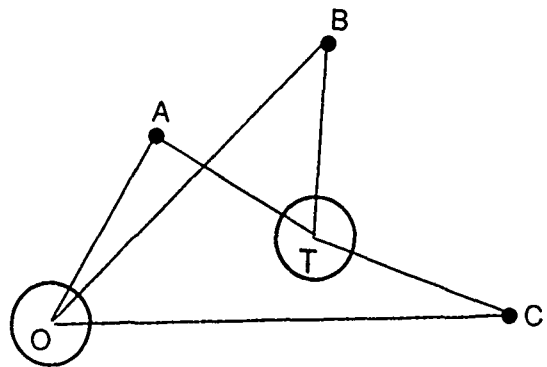


Figure 6: Homing problem with three landmarks.

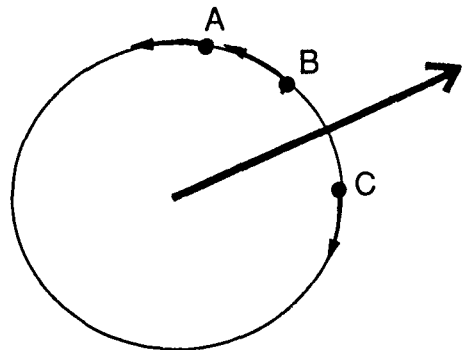


Figure 7: Landmarks slide around horizon circle as the robot moves.

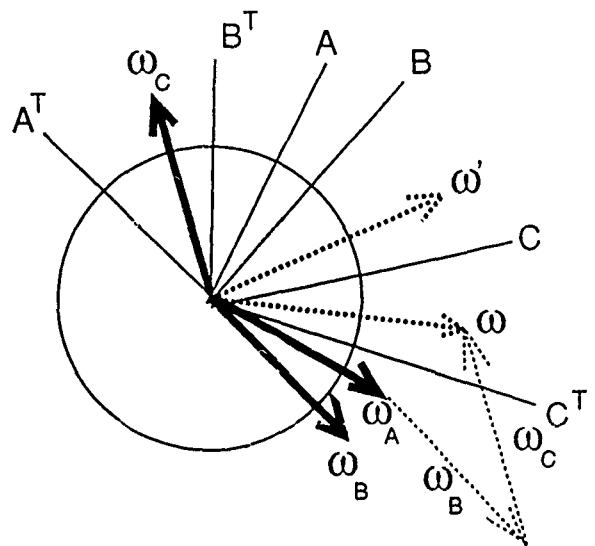


Figure 8: How a robot uses landmarks to estimate homing direction.

where ϕ_i is the offset of the i^{th} characteristic point and θ_i is its bearing.

As well as determining a direction to move, the algorithm must also choose the distance to move. For each target signature, the robot is moved 0.6 feet on the first incremental homing step, 0.4 feet on the second incremental homing step and 0.3 feet on subsequent incremental homing steps. We choose initial large values so that the robot makes greater progress on the first two steps; the final small values ensure that the robot does not overshoot the target location by much. The homing process for a given target location signature stops when the current direction the robot moves, $\omega(\tau)$, differs sufficiently from the previous direction it moved, $\omega(\tau - 1)$. When this happens, it is assumed that the robot has overshoot the target location. In our system the stopping criterion is $|\omega(\tau) - \omega(\tau - 1)| > 40^\circ$. Of course, many other stopping criteria are possible, such as a threshold on the degree-of-match between current and target location signatures.

As part of the matching process, we estimate any (unintentional) rotation of the robot's body that may have occurred since the time the target signature was acquired. We call this rotation the *deviation*. Unless the estimated deviation is subtracted from the offsets of the characteristic points, the robot's homing movement will be biased in the direction of the deviation. In practice, the deviation is small — less than 3° . A forthcoming technical report will give details of how we estimate deviation [7].

6 Demonstration of the Homing Algorithm

We tested the algorithm by taking a sequence of hemispherical images in a hallway. There were 17 images in all, taken at target locations spaced about 1 foot apart. We placed the robot 1 foot from the first target location; its goal was to home to each target location in sequence. The robot was able to traverse this path successfully and reach the final target location.

Figure 9 shows the hemispherical image of the world as seen from the first target location. Figure 10 shows the view at the first incremental homing step towards the first target location. We have superimposed a circular coordinate system over each image, with the origin of the coordinate system at the center of the cross-hairs. Absolute angles are measured in the standard mathematical convention: $\theta = 0^\circ$ is the x-axis, and angles increase in a counter-clockwise direction. Each tick-mark in the annulus marks the part of the image that was radially sampled to produce one intensity component of a location signature.

We show the target image and the images from each homing step to show the features the algorithm picked out as characteristic points and the matches it made. The fifteen characteristic points extracted from the location signature at each of the five incremental steps are labeled in Figures 8 through 12 as *A* through *O*; the subscript on each label is the number of the step in which

the characteristic point was extracted. Since the algorithm picks out different sets of characteristic points at each step, it is important to realize that an alphabetic label may indicate different structural features in different steps. For example, the edge of a door is labeled H_1 in Figure 10, but the same edge is labeled J_2 in Figure 11. Figure 9 shows what the characteristic points from all five steps matched in the target signature. For example, landmark *C* in step 2, *A* in step 3, *A* in step 4, and *B* in step 5 all matched to the edge of a door at bearing $\theta = 32^\circ$ in the target image.

The performance of the algorithm at each step is shown in Tables 1 through 5. Each row in these tables shows the label of the characteristic point (C.P.), its bearing (θ), the angular offset to the matching point in the target image (ϕ), and the correlation between the characteristic point and its matching feature (ρ , where the lower the value, the better the match).

The tables also show whether the characteristic point was judged to be a good characteristic point (C.P. OK?) and whether the algorithm matched the characteristic point to the correct feature in home image (Match OK?). We judged a characteristic point to be good if it was within one pixel of an obvious structural feature such as the edge of a door or a sudden reflectance change. If no such feature was apparent, we judged the characteristic point to be bad. Thus, characteristic point O_1 in Figure 10 was judged to be good because we can readily see the feature that caused it: a large area of dark bulletin board next to an area of white paper. Characteristic point A_1 , however, was judged to be bad because there is no obvious structural feature associated with it. We judged a match to be good if the characteristic point matched to a point within one pixel of the correct point in the home image.

In general, the algorithm picked out good characteristic points. In steps 1 through 4, 13 out of 15 characteristic points were good; in step 5, 9 out of 15 were good. In matching, however, the algorithm was less successful. Table 6 summarizes the algorithm's performance in matching characteristic points. On the average, the algorithm matched characteristic points to correct points in the location signature only two-thirds of the time. Nevertheless, this was sufficient for homing.

Looking more closely at Table 6, we see that the algorithm made more good matches than bad in steps 2 through 5, but in step 1 it made *fewer* good matches than bad. The robot moved in the correct direction in steps 2 through 5 because the effects of the good matches outweighed the effects of the bad. How, then, did the robot manage to move correctly in step 1, where there were more bad matches than good?

In general, the offsets and bearings of bad matches should be randomly and uniformly distributed in the interval 0° to 360° . If this is true, the expected value of the vector sum of those homing vectors derived from bad matches will be the zero vector. Since the bad matches can be expected to contribute little to the final homing vector, the direction of the final homing vector should be determined by the vectors from the good matches.

This intuition is supported by the results in Table 7.

For each step, we sum the homing vectors contributed by the good matches into a resultant vector; we do the same for the bad matches. The resultants are decomposed into their x and y components, where the positive y direction lies up the corridor towards the target location and the positive x direction is 90° clockwise from the positive y direction. We can see that in the first step the bad matches conspire to drive the robot away from the target location (in the negative y direction) while the good matches drive the robot towards the target. Even though there are more bad matches than good in step 1, their y -component is smaller than the y -component of the good matches because they tend not to be correlated. Thus, the vectors from the bad matches partially cancel each other out, as expected.

For steps 2 through 4 there are more good matches than bad, so it is not surprising that their y -component is larger. The very small, negative y -component for the good matches in the step 5 is because the robot has just overshot its target location slightly, causing step 5 to be the last step in homing to this target location.

The homing problem can be decomposed into two parts: the "correspondence problem" and using the correspondences to compute the homing vector. We believe the performance of the matching algorithm in solving the correspondence problem can be improved in future work. The main contribution of this paper, however, is the novel way that the system uses the correspondences to navigate. The apparent robustness of the system is impressive: not only did it manage to home correctly when many matches were incorrect, but it homed to seventeen target locations in a row. Since the robot needed (on average) about four incremental homing steps to reach each target, the robot behaved correctly on about 60 to 70 homing steps in a row.

As an aside, we would like to mention that we weighted the homing vector for each characteristic point by a heuristic weighting function in the actual experiment. This function was $\frac{S|\phi|}{\rho}$, where S is the characteristic points sparseness (i.e., distance from other landmarks), ϕ is its offset, and ρ is correlation of its match. The intuition behind sparseness was that characteristic points that were close to each other would unduly bias the robot to move in their associated direction. To compensate, sparse characteristic points — which were distant from other characteristic points — were given more weight. Since characteristic points with large offsets are responsible for most of the discrepancy between a target signature and the current signature, it also seemed reasonable to try to reduce the larger offsets first. Hence, those characteristics points were given more weight. Finally, since large correlation values imply poor matches, we used the reciprocal of the correlation value in the weighting function. Since this weighting function neither helped nor hindered the performance of the algorithm, we will not discuss it further here. A forthcoming technical report will give more details [7].

7 Summary and Discussion

We have argued that image-based landmark navigation is a feasible alternative to navigation approaches that maintain three-dimensional models of the world. This paper describes an approach that divides large-scale navigation tasks into a sequence of small-scale navigation tasks that are solved by local, image-based homing. Our homing algorithm uses compact location signatures acquired by a novel 360° imaging system. In addition, landmark information is acquired in a natural and straightforward way that does not involve acquiring three-dimensional information. We have described our image-based homing algorithm and have demonstrated it on a mobile robot for a typical short-range navigation task.

In future work we will be trying to improve the robustness of the homing algorithm and to extend its range. Our experiment showed that the homing algorithm correctly picked out characteristic points, but mismatched many points. Clearly we need to improve the matching part of the algorithm. Whatever the shortcomings of the current implementation of the homing algorithm, it was still able to successfully navigate using real-world images. Ultimately, image-based local homing might be used to create a full-blown navigation system that can autonomously acquire a qualitative spatial map of its environment for robust, goal-oriented navigation.

8 Acknowledgements

This article describes research done at the Department of Computer and Information Science, University of Massachusetts, Amherst. This work would not have been done without the efforts of Jonathan Hartl, Gary Wu and Zhongfei Zhang. They helped debug and test the navigation system, getting it to the point where we could painlessly run experiments; they also ran many experiments. We also thank David Ehrenberg and Valerie Cohen for their help in keeping the robot alive and Robert Heller for his help in pulling all the software together.

References

- [1] Gilad Adiv.
Interpreting Optical Flow.
PhD thesis, University of Massachusetts/Amherst. 1985.
- [2] Nicholas Ayache and Oliver D. Faugeras.
Maintaining representations of the environment of a mobile robot.
IEEE Transactions on Robotics and Automation 5(6): 804-819, December 1989.
- [3] Thomas M. Breuel.
Adaptive model base indexing.
In *Image Understanding Workshop 1989*, pages 805-814. Morgan Kaufman, 2929 Campus Drive, San Mateo, California. 1989.
Proceedings of a workshop sponsored by DARPA.
- [4] Ernest Davis.
Representing and Acquiring Geographic Knowledge.
PhD thesis, Yale University, 1984.

- [5] Ernst Dieter Dickmanns and Volker Graefe.
Dynamic monocular machine vision.
Machine Vision Applications 1(4):223-240, 1988.
- [6] Claude Fennema, Allen R. Hanson, Edward Riseman, J. Ross Beveridge and Rakesh Kumar.
Model-directed mobile robot navigation.
To appear in *IEEE Transactions on Systems, Man and Cybernetics*.
- [7] Jia-Wei Hong, Xiaonan Tan, Brian Pinette, Richard Weiss and Edward M. Riseman.
Image-based Navigation Using 360° Views.
Department of Computer and Information Science,
University of Massachusetts at Amherst. 1990.
Forthcoming technical report.
- [8] Berthold K. P. Horn.
Relative orientation.
In *Image Understanding Workshop 1988*, pages 826-837. Morgan Kaufman, 2929 Campus Drive, San Mateo, California. 1988.
Proceedings of a workshop sponsored by DARPA.
- [9] R. A. Jarvis and J. C. Byrne.
An automated guided vehicle with map building and path finding capabilities.
In Robert C. Bolles and Bernard Roth, editors, *4th International Symposium on Robotics Research*, pages 497-504. MIT Press, Cambridge, Massachusetts. 1988.
- [10] Rakesh Kumar and Allen R. Hanson.
Robust estimation of camera location and orientation from noisy data having outliers.
In *IEEE Computer Society Workshop on Interpretation of Three-dimensional Scenes*, pages 52-60. IEEE Computer Society Press. 1989.
- [11] Randal C. Nelson.
Visual homing using an associative memory.
In *Image Understanding Workshop 1989*, pages 245-262. Morgan Kaufman, 2929 Campus Drive, San Mateo, California. 1989.
Proceedings of a workshop sponsored by DARPA.
- [12] Hisashi Suzuki and Suguru Arimoto.
Visual control of autonomous mobile robot based on self-organizing model for pattern learning.
Journal of Robotic Systems 5(5):453-470. 1988.
- [13] S. Ullman.
The Interpretation of Visual Motion.
MIT Press, Cambridge, Massachusetts. 1979.
- [14] Wai K. Yeap.
Towards a computational theory of cognitive maps.
Artificial Intelligence 34:pages 297-360, 1980.
- [15] Jiang Yu Zheng and Saburo Tsuji.
Panoramic representations of scenes for route understanding.
In *Tenth International Conference on Pattern Recognition*, pages 161-167. IEEE Computer Society Press. 1990.
- [16] D. Zipser.
Biologically plausible models of place recognition and goal location.

In David E. Rumelhart, Jay L. McClelland and the PDP Group, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 2: Psychological and Biological Models*, pages 432-471. MIT Press, Cambridge, Massachusetts. 1986.

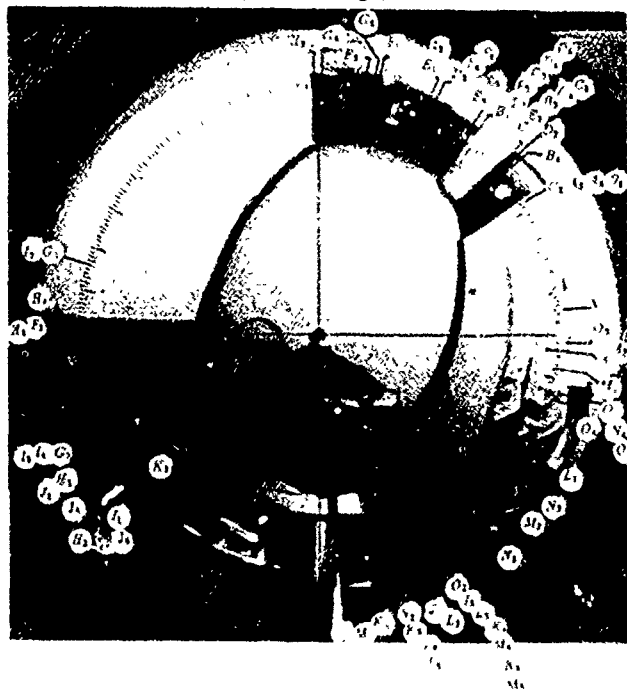


Figure 9: Hemispherical image of hallway as seen from target location. Labels show the places matched by the characteristic points from all five incremental steps.

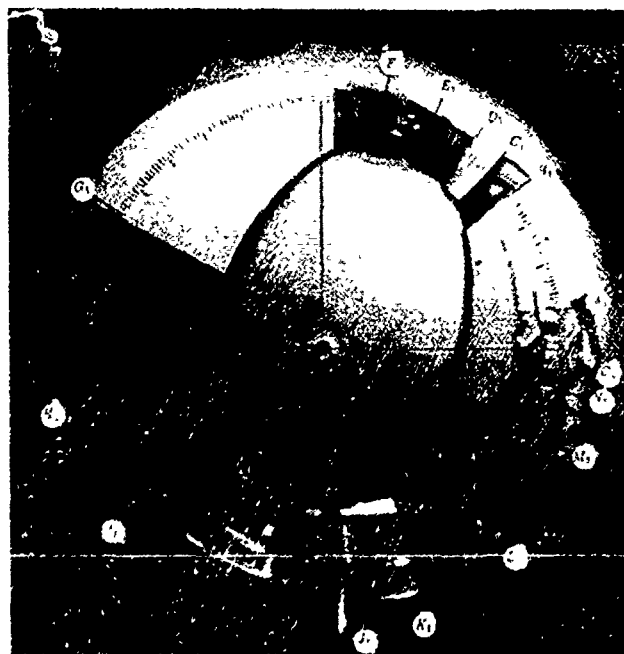


Figure 10: Hemispherical image of hallway as seen at the start of the first incremental step. Labels indicate the characteristic points found on this step.

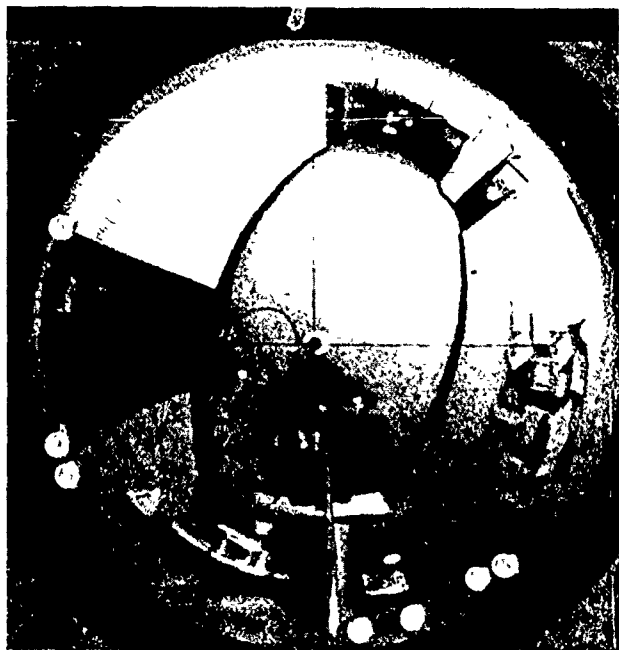


Figure 11: Hemispherical image of hallway as seen at the start of the second incremental step. Labels indicate the characteristic points found on this step.

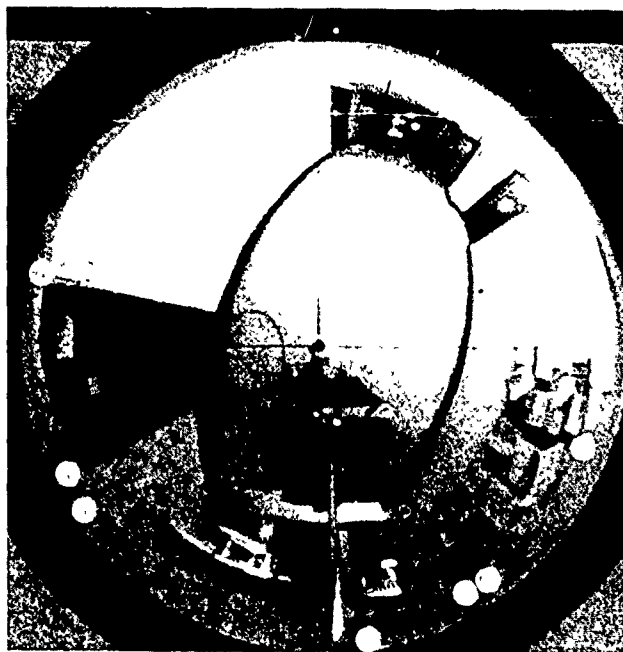


Figure 13: Hemispherical image of hallway as seen at the start of the fourth incremental step. Labels indicate the characteristic points found on this step.

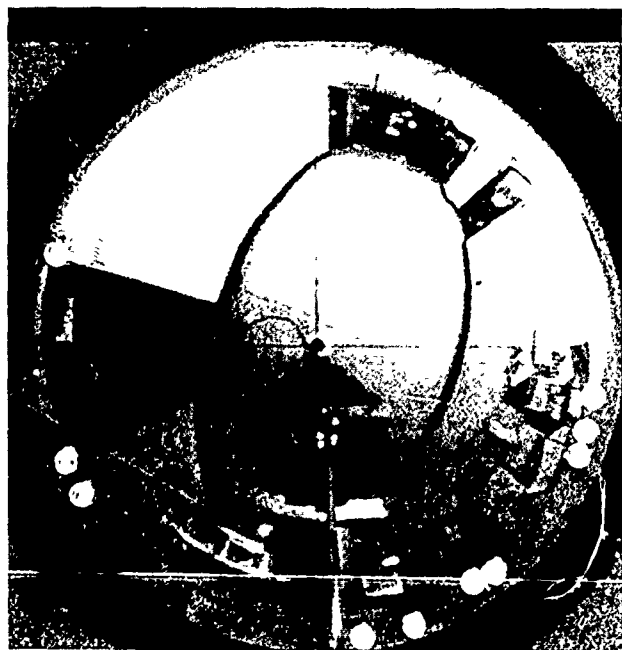


Figure 12: Hemispherical image of hallway as seen at the start of the third incremental step. Labels indicate the characteristic points found on this step.

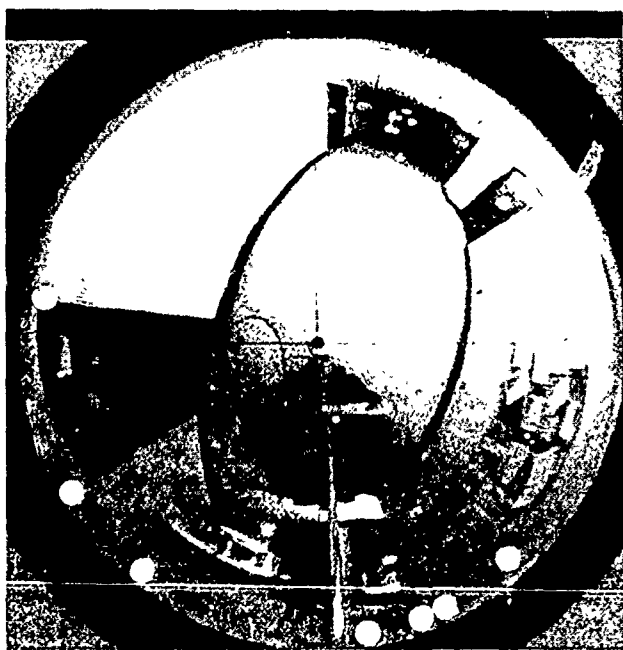


Figure 14: Hemispherical image of hallway as seen at the start of the fifth incremental step. Labels indicate the characteristic points found on this step.

INCREMENTAL HOMING STEP 1					
C.P.	Bearing θ	Offset ϕ	Correl. ρ	C.P. OK?	Match OK?
A ₁	9	-13	40	n	n
B ₁	38	13	20	y	n
C ₁	45	-3	29	y	y
D ₁	52	-1	10	y	y
E ₁	63	1	60	y	y
F ₁	75	1	62	y	y
G ₁	148	16	197	y	n
H ₁	194	15	58	y	y
I ₁	220	-2	51	y	n
J ₁	279	15	83	y	n
K ₁	292	-10	36	y	y
L ₁	315	14	51	n	n
M ₁	338	14	52	y	n
N ₁	350	16	63	y	n
O ₁	355	-8	56	y	n

Table 1: Characteristic points and their matching features seen at step 1.

INCREMENTAL HOMING STEP 4					
C.P.	Bearing θ	Offset ϕ	Correl. ρ	C.P. OK?	Match OK?
A ₄	32	0	18	y	y
B ₄	39	1	45	y	y
C ₄	41	1	28	y	y
D ₄	49	2	38	y	y
E ₄	51	2	82	y	y
F ₄	62	1	39	y	y
G ₄	74	15	80	y	n
H ₄	166	10	91	y	y
I ₄	207	1	83	y	y
J ₄	213	1	17	n	n
K ₄	280	16	100	y	n
L ₄	303	-10	20	y	y
M ₄	306	-10	35	y	y
N ₄	340	7	49	y	n
O ₄	354	-9	51	n	n

Table 4: Characteristic points and their matching features seen at step 4.

INCREMENTAL HOMING STEP 2					
C.P.	Bearing θ	Offset ϕ	Correl. ρ	C.P. OK?	Match OK?
A ₂	3	-7	46	n	n
B ₂	11	-15	11	y	y
C ₂	35	-3	27	y	y
D ₂	42	-1	78	y	y
E ₂	43	-1	80	y	y
F ₂	51	0	4	y	y
G ₂	63	1	70	y	y
H ₂	76	15	59	y	n
I ₂	156	8	195	y	n
J ₂	201	8	58	y	y
K ₂	208	9	10	n	n
L ₂	279	15	88	y	n
M ₂	291	-9	26	y	y
N ₂	308	-15	42	y	y
O ₂	311	-15	50	y	y

Table 2: Characteristic points and their matching features seen at step 2.

INCREMENTAL HOMING STEP 5					
C.P.	Bearing θ	Offset ϕ	Correl. ρ	C.P. OK?	Match OK?
A ₅	1	-5	38	y	y
B ₅	31	1	12	y	y
C ₅	41	1	76	y	y
D ₅	49	2	46	y	y
E ₅	51	2	92	y	y
F ₅	62	1	36	y	y
G ₅	76	1	41	y	y
H ₅	171	6	43	y	y
I ₅	210	-2	100	n	y
J ₅	231	-13	7	n	n
K ₅	280	16	110	y	n
L ₅	302	-9	19	n	n
M ₅	305	-9	32	n	n
N ₅	322	-10	27	n	n
O ₅	353	-6	15	n	n

Table 5: Characteristic points and their matching features seen at step 5.

INCREMENTAL HOMING STEP 3					
C.P.	Bearing θ	Offset ϕ	Correl. ρ	C.P. OK?	Match OK?
A ₃	33	-1	17	y	y
B ₃	42	0	25	y	y
C ₃	50	1	22	y	y
D ₃	62	1	44	y	y
E ₃	76	3	52	y	y
F ₃	162	15	30	y	y
G ₃	204	4	20	y	y
H ₃	211	4	11	n	n
I ₃	280	16	82	y	n
J ₃	290	-8	25	y	y
K ₃	305	-12	23	y	y
L ₃	308	-12	43	y	y
M ₃	339	-15	27	y	y
N ₃	343	-15	17	y	y
O ₃	351	7	24	n	n

Table 3: Characteristic points and their matching features seen at step 3.

Step	Correct Matches		Incorrect Matches	
	Number	(%)	Number	(%)
1	6	(40)	9	(60)
2	10	(66.67)	5	(33.33)
3	12	(80)	3	(20)
4	10	(66.67)	5	(33.33)
5	9	(60)	6	(40)
Total	47	(62.67)	28	(33.33)

Table 6: Comparison of correct matches to incorrect matches at each step and over all steps.

Step	Correct Matches		Incorrect Matches	
	x	y	x	y
1	0.2385	2.2048	0.0752	-1.5849
2	-0.2326	5.0352	-1.0207	2.1726
3	3.9096	5.9229	-1.4217	-0.9025
4	5.6806	1.0645	-0.9134	0.1101
5	5.7486	-0.3599	1.2575	2.7526

Table 7: Comparison of components of homing vector contributed by correct matches to those contributed by incorrect matches at each step and over all steps.

Reactive and Preplanned Control in a Mobile Robot

Monnett Hanvey Soldo

Department of Computer Science
Columbia University
New York, NY 10027

Abstract

We describe a novel organization for robot navigation that combines reactive, stimulus-response control with cognitive, planned behavior government. The result is robust, flexible, autonomous, real-time robot control; this result has been demonstrated on a mobile robot that explores the peopled hallways of a large office building.

1. Introduction

The 1989 AAAI Spring Symposium on Robot Navigation was the scene of a heated debate — or panel discussion — between the reactivists and the planners. On the one side, the reactivists, citing demonstrated successes in robot mobility,^{[1] [2] [3] [4]} advertised independence from specific world models: behavior should be defined in terms of response to significant stimuli. (If memory serves, the battle cry was "we don't need no stinking maps!") On the other, the planners protested that there's no way to achieve intelligent, autonomous behavior without representation and planning. They're both right, of course.

Among mobile roboticists, most of the interest in reactive control seems to have grown out of Brooks' subsumption architecture^[1]. It is an approach strongly supported by psychology^[6] and biology^[6], and one which has been successfully applied by control theorists for years. (Witness the autopilot.) Path planning is a more traditional AI approach to navigation. The robot is given a symbolic map of its environment that can be searched for a desirable path. Often these planners are hierarchical: a rough, high-level plan is constructed to guide the robot, and the details of each path segment are filled in depending on local conditions^[7].

Reactive control has the advantage of timely response, and is essential in some domains. (The greatest enemy to an athlete is conscious intervention^[8].) But the mechanisms of reactive control are highly specialized, and more general world knowledge and reasoning ability is required to address a variety of situations. In this paper we present an architecture for navigation that combines reactive and planned control. This architecture has been implemented and demonstrated on a mobile robot that explores the hallways of an office building.

2. The Robot

Ours is a three-wheeled, indoor robot, about two feet wide and four feet tall. Two of its three wheels are

differentially driven, while the third (front) is a castor. All power and processing are onboard, to speed response time and to allow the robot to roam long distances untethered.

Robot trajectories are specified in terms of desired forward motion and desired change in orientation: We use quintic (fifth-order) polynomials, after Andersson^[9], to specify robot trajectories for forward motion and turn, and then integrate the two trajectories and translate them to wheel velocities in each servo cycle. Top speed for the robot is five feet per second, but we prefer running it at about 1.5 fps.

The robot carries eight ultrasonic sensors, three along each side and two in the front, for detecting obstacles. (These sensors are mounted about three feet off the ground.) It has odometry on each of the two driven wheels. And it has a single camera, mounted in front and facing forward, at the level of the sonar sensors; images are processed through special-purpose hardware for speed. The transparent cylinder between the battery layer and the processing layer is reserved for a laser rangefinder to be installed in the future.

3. Control Structure

Control of the robot is distributed among a set of *behavior experts* that tightly couple sensing and action. (See figure 1.) The interesting problems here are to select a small but useful set of parameters to describe the world — parameters that can directly affect robot behavior — and to devise strategies for sensing them quickly. The choice of parameters is less obvious for a mobile robot than for an autopilot, given a more complicated world (Note that it depends on behavior goals: front clearance is only important if the robot isn't supposed to run into anything.) Sensing strategies may involve discarding most of the sensed data to extract only what's needed.

The resulting behavior experts incorporate specialized processing to sense and to act; they can be viewed as sophisticated servo-controllers. The distribution of control, together with fast sensor processing, allows real-time response as the robot navigates in a changing environment.

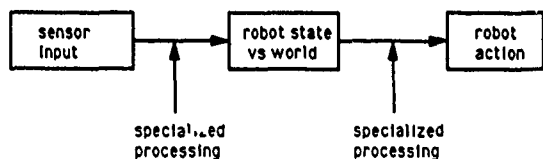


Figure 1. Form for a behavior expert.

If a group of behavior experts are all active at the same time, setting various behavior parameters, together they will produce a global behavior for the robot. To introduce some order into this scheme, we collect behavior experts to define robot behaviors. (Figure 2.) A specific set of behavior experts can be activated by selecting the appropriate (global) robot behavior.

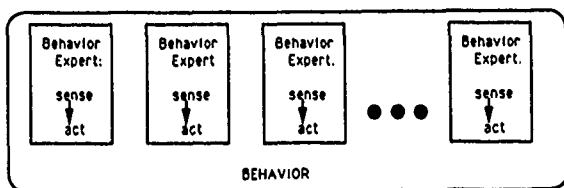


Figure 2. Form for a robot behavior.

Some of the behavior experts within a behavior may in fact be boundary experts (daemons), whose function is to trigger a change in behavior under specific conditions. These boundary experts define relationships among the various robot behaviors (figure 3), and those relationships can be used in assembling the behaviors into an AI plan to direct the robot.

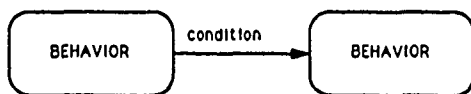


Figure 3. Behavior relationships.

In the sections that follow we describe the specific behavior experts, behaviors, and plans that control our robot.

4. Sensing for Action

Our robot navigates indoors, in a hallway environment. It has no prestored map of the world, and it is not directed to follow any predetermined trajectory. The robot's mission is to roam the hallways — to "explore" — without running into anything. This mission suggests a few basic behaviors. The robot is supposed to (and does) travel along corridors, turn at intersections, and retreat from dead ends.

Here we look at one of these behaviors: travelling along a single corridor. We identify two of the behavior experts that control the robot within a hall.

4.1 Moving Forward

We want the robot to roll from one end of the hallway to the other. In a static environment, we could generate one long trajectory to take the robot all the way down the hall, and then worry only about getting the robot to execute it accurately. But our robot's world is not static; it is

populated by people who wander past, largely ignoring the robot as it rolls along. (This is interesting: one would think that an untethered robot roaming the halls would be exciting, an attention-getter, something you don't see every day.) Collisions are bad for public relations and bad for the robot, so the robot has to be prepared to stop if necessary to avoid them.

In a dynamic world it is pointless to specify trajectories that take the robot very far, and equally pointless to specify trajectories long before they are executed. Our robot generates trajectories on the fly, based on its current situation, roughly three times per second. A new trajectory will interrupt the one that's already executing, so that the repeated trajectory generation produces continuous robot motion.

Trajectories are at most a few feet in length; shorter if an obstacle is detected ahead. (Down to a minimum length of zero, a full stop, when an obstacle is detected less than one foot away.) To find obstacles we use two forward-pointed ultrasonic sensors.¹ In these sensors the shorter measurements are generally the accurate ones — most of the errors we see are due to reflections — so the shorter of the two measures is used.

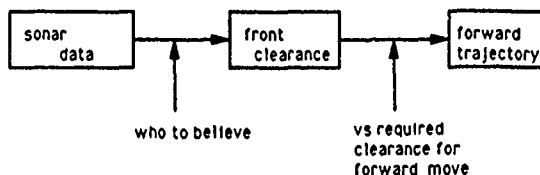


Figure 4. Forward motion expert.

This behavior expert, drawn in figure 4, links data from the forward-looking ultrasonics to robot motion. The robot moves forward if the area ahead is clear, but stops if its path is blocked.

4.2 Staying Straight

As it rolls down the hall, the robot should stay straight (aligned with the length of the hall), not only for efficiency but also to insure that it can sense the walls on either side. (Side-mounted ultrasonic sensors are the robot's only means of sensing obstacles to its sides; if it is not oriented roughly straight, reflections may prevent it from detecting the walls.) To be safe it should also stay near the center of the corridor. (The robot is two feet wide, and it cannot sense obstacles to its sides closer than about nine inches, this means that only a couple of feet in the center of a six-foot hall are reasonably safe.)

1 We should note that the resolution of these sensors is so poor that we use them to find (hopefully) the distance to the nearest obstacle practically *anywhere* across the width of the hallway. Consequently we do not (cannot) attempt obstacle avoidance, but instead we just stop the robot if the obstacle is too close. Also note that one can never guarantee collision avoidance in a dynamic environment, for the same reason that most of us can't avoid speeding bullets. The best we can do is insure that we can avoid objects moving within specified velocity constraints (depending on the robot's sense/react time) if the sensors and motor control work properly.

We associate with each hallway its own local coordinate system: z is oriented with the hall, and x across it, with $x = 0$ down the middle of the hall. Robot position is specified by the triple (x, z, θ) , where θ represents orientation. Our desire that the robot stay straight is expressed as a preference for $\theta = 0$; staying near the hallway center implies keeping x near 0.

As the robot moves forward, errors in both x and θ can be corrected by changing robot orientation. For each trajectory generated — at least three per second — the robot needs an accurate estimate of its position, especially x and θ . The robot has odometry on each of two differentially driven, load bearing wheels. (Roughly 24000 ticks per foot.) The encoder counts are checked frequently — almost two hundred times per second — and they can be integrated to provide an updated estimate of robot position. It is well known, however, that errors introduced by wheel slippage, uneven terrain, imperfect wheels, *etc.* will accumulate, in time making the odometry-based estimate useless. In fact, there are errors built into the estimate itself. encoder values are only read at discrete intervals, and there's no way to know what went on between readings.²

The robot cannot zero out the odometry errors occasionally by recognizing known landmarks: accurate position information has to be available more often than "occasionally," and anyway this implies a prestored map, something our robot doesn't have. Instead, our robot selects its own "landmarks" on the fly. The robot's two-dimensional world is punctuated by vertical edges produced where the walls are broken for doorways, corners, *etc.*; these edges make great position cues, and they are used as landmarks by the robot.

The robot is equipped with a camera facing forward and hardware to extract strong vertical edges from the raw image data. (The robot does not have any prestored map suggesting where to find edges; it just uses whatever edges it sees.) Using a filter-based mechanism that combines vision and odometry — to be described in a future report³ — the robot tracks the positions of these edges over time, and uses their apparent motion to track its own position. The estimator runs at 60 Hz, to provide accurate an position estimate at (practically) any time.

For each forward trajectory generated (above), the robot generates a trajectory to achieve a correction of $\Delta\theta$ in orientation. The $\Delta\theta$ is a combination of the corrections suggested by x and θ : θ suggests a correction of $-\theta$, and x suggests a correction that takes it toward the center. (The combination is a weighted average; we use equal weights of 0.5. Changing the weights can produce some interesting results, with large enough changes affecting the robot's "personality.")

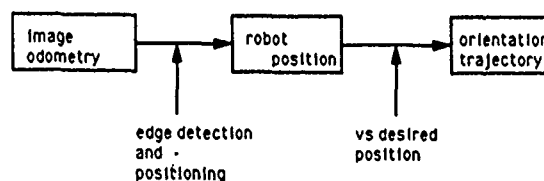


Figure 5. Orientation expert.

This behavior expert (figure 5) discards most of the data from each image, extracting only what it needs and can process quickly. Again, there is a direct link between sensor data and robot motion: as it moves forward the robot corrects for any deviation from straight and level.

4.3 Commentary

At this point some comments are in order regarding reactive behavior. Behavior experts are highly specialized devices, tailored to a particular type of environment, specific robot sensors and mobility, and specific behavior goals. Change one of these parameters and we'll have to define new experts. Change the environment and the experts can be fooled; change the robot or its sensors and they're obviously no longer valid. (More on changing behavior goals in a minute.) This specialization is also evident in biological matched filters, but Wehner^[6] has commented that it is not necessarily bad: "to us, [the filters] might cover only some partial aspects of the more comprehensive geometrical problems we would like to envisage; but to the animals they are always the full solutions to the very problems with which they must contend."

We claim that in fact behavior experts can be quite general. There is a reasonable argument that the shortest description of a set of data is the most general, in that it captures the structure — rather than the specific details — in the data. There is a structure inherent in our robot's environment, essentially a network of one-dimensional spaces. That structure, with consideration of the robot's abilities and task, was the source of our behavior experts. In a handful of behavior parameters these devices capture what is common to navigation/exploration behaviors in hallway environments.⁴ (In some sense our experts procedurally encode the generic environment information described by Kriegman^[11].) The robot who uses these experts could be started anywhere in any hallway world, without any predefined map, and it would go happily about its business.

5. Behaviors and Planning

Control of the robot is distributed among a set of behavior experts, but together a group of behavior experts produces a global behavior for the robot, simply by virtue of their coexistence. We organize this potential chaos by grouping the experts into robot behaviors, *e.g.*, going down a hall. The activation of experts is thus governed at a higher level by the activation of robot behaviors. (Going down the hall requires the experts for collision-free forward motion, for staying centered, and others; see figure 6.)

2. Another way to see the problem: from the two measures provided by the encoders — forward motion and spin — we need three Δx , Δz , and $\Delta\theta$.

3. The technique is similar to that of Crowley^[10] in that the same data is used to update the model and to correct position.

4. We make no claim that the parameters we have used are by any means the ideal ones, for our application or for anyone else's. They were arrived at through experimentation, and are presented here to demonstrate the feasibility of the approach.

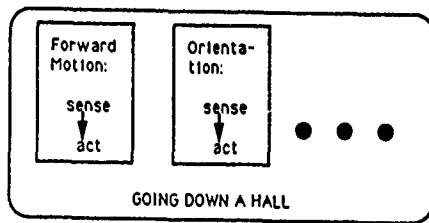


Figure 6. Behavior: Going down a hall (partial description).

Those high-level behaviors can be assembled into an AI "plan" to direct the robot. Currently our robot has one such plan, prescribed by us, that directs it to travel along corridors, turn at intersections, and retreat from dead ends. (We have not discussed turning and retreating here.) This plan takes the form of a finite state machine; a subset of that machine appears in figure 7. (We have left out states that are only interrupt handlers — for example, what to do if someone blocks your path while you are turning — in the interest of making the diagram legible.)

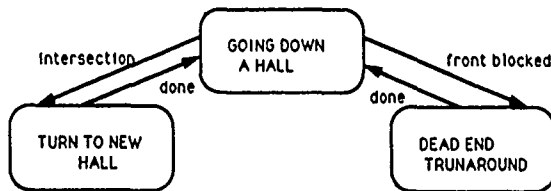


Figure 7. Finite state machine for behavior control.

The robot does not now construct plans on its own; given a representation and a goal, however, planning should be a straightforward extension. In any plan, behavior experts not only direct activities within a behavior but also trigger transitions from one state to the next.

5.1 Commentary

Goal-directed behavior requires a map. Nobody builds a map by storing a whole sequence of raw image data. The real reason that this doesn't happen is not memory limitation but rather utility: the aim is to store information so that it will be useful (without excessive processing) in the future, so that at a moment's notice it can be retrieved and applied. But if the abstraction is not done carefully, it may obscure whatever was interesting in the data, again making the representation less valuable. We propose that robots ought not represent the complete spatial configuration of the world, but instead should know where they can go and what they can do therein. We are now experimenting with representing the world in terms of robot behaviors

6. Demonstration of Success

Our robot has explored its hallway environment many times, using the control architecture described here. The system has proved robust, even in busy corridors. And it is flexible: the robot can explore from any position in any hallway environment. The longest of its journeys covered more than 500 feet on two different floors (We helped it into and out of the elevator.)

Figure 8 shows a trace of the robot's movements in a single hallway. The positions recorded (arrow represents θ) are its own estimates; these estimates are very accurate. (Note the scale/path length.) A future report will present a new technique for recording ground truth in such experiments. Hallway walls are at $x=3$ feet and $x=-3$ feet.

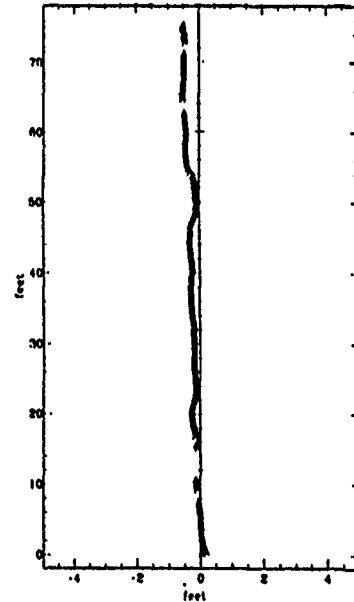


Figure 8. One robot path. Positions are as estimated by the robot.

7. Conclusion

We have presented an organization for robot navigation that combines reactive and preplanned control, and has been demonstrated on our robot. The system shares features of reactive control with others^{[1][4][8][12]}, but it is unique in its integration of planning with reactive control. (A similar approach, however, was recently suggested by Arkin^[5].) We believe that the organization could be effective for non-mobile robots as well.

Our current interest is in the implications of this organization for robot representation. Given the emphasis on behavior experts, it is likely that a behavior-based representation — what sensor input should instantiate the end-of-hall behavior? — will be more useful than a straightforward spatial representation. The representation should be designed to steer the robot's actions within the world, not necessarily to represent the spatial layout of that world as completely as possible.

Acknowledgements

My thanks to John Kender, John Jarvis, Russ Andersson, Peter Allen, Terry Boulton, and Bob Lyons. (They know why.) The robot described exists at AT&T Bell Laboratories in Holmdel, NJ.

REFERENCES

1. Rodney Brooks, "A Layered Intelligent Control System for a Mobile Robot", *IEEE Journal of Robotics and Automation*, Vol. RA-2, No. 1, pp. 14-23, April 1986.
2. Ronald C. Arkin, "Motor Schema Based Navigation for a Mobile Robot: An Approach to Programming by Behavior", *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, pp. 264-271.
3. David W. Payton, "An Architecture for Reflexive Autonomous Vehicle Control", *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, pp. 1838-1845.
4. Ernst D. Dickmanns, "Active Vision through Prediction Error Feedback", *NATO Advanced Study Institute on Active Perception and Robot Vision*, Martea, Italy, 1989.
5. Ronald C. Arkin, "Towards the Unification of Navigational Planning and Reactive Control", *Working Notes, AAAI Spring Symposium on Robot Navigation*, Stanford University, 1989.
6. Rüdiger Wehner, "'Matched Filters' — neural models of the external world", *Journal of Comparative Psychology A* 161:511-531, 1987.
7. Yoshimasa Goto and Anthony Stentz, "The CMU System for Mobile Robot Navigation", *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, pp. 99-105.
8. Lawrence Shainberg, "Finding 'The Zone'", *The New York Times Magazine*, April 9, 1989.
9. Russell L. Andersson, *A Robot Ping-Pong Player: Experiment in Real-Time Intelligent Control*, MIT Press, 1988.
10. James L. Crowley, "World Modelling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging", *Proceedings of the 1989 IEEE Conference on Robotics and Automation*, pp. 674-680.
11. David J. Kriegman, Thomas O. Binford, and Thilaka Sumanaweera, "Generic Models for Robot Navigation", *DARPA Image Understanding Workshop*, 1988, pp. 453-460.
12. Jonathan Hudson Connell, "A Colony Architecture for an Artificial Creature", *MIT AI TR-1151*, 1989.

General Routing on the Lowest Level of the Image Understanding Architecture

Martin C. Herbordt and Charles C. Weems

Computer and Information Sciences Department, University of Massachusetts at Amherst
Amherst, Massachusetts 01003

David B. Shu

Hughes Research Laboratories, 3011 Malibu Canyon Road, Malibu, CA. 90265

Abstract

SIMD mesh connected computers have been found to be very useful in many applications, such as those often found in image processing and matrix arithmetic, where the communication among PEs is local, regular, or both. Low efficiency results, however, when the communication patterns are irregular or sparse, a situation we have found to exist in many computer vision applications. This paper describes a routing primitive which makes significant progress towards solving that problem for the CAAPP, a SIMD mesh connected computer enhanced with the broadcast capability of the coterie network. We show that general routing on the CAAPP can be executed with simplicity to the user and performance similar to that of a dedicated network. We present experimental results from several classes of permutations as well as from some common machine vision applications.

1 Introduction

One class of architectures that has proven popular for use in image processing and other domains that map readily onto fine-grained parallel computation is the SIMD mesh connected computer (SMCC). Some of the advantages of mesh connected topology are that it is regular and easy to lay out on a chip, has high bandwidth for data movement along its dimensions and low latency for local transfers; some disadvantages are its large diameter and limited bandwidth when the data movement is not regular. It is therefore apparent why SMCC architectures have found their greatest success in two areas: the first is in modeling certain physical phenomena, such as images, which map naturally onto a mesh and for which many relationships are local; the second is in executing regular mathematical computations, such as matrix operations. Conversely, mesh connected topologies are least efficient when confronted with computations, many of which occur in computer vision, that require communication between distant PEs, especially when the communication

pattern is sparse and/or cannot easily be described in terms of nearest neighbor moves.

In this paper we present the ROUTE primitive, a collection of routing algorithms for the Content Addressable Array Parallel Processor (CAAPP)[15], which allow many more classes of interprocessor communication to be executed efficiently than could otherwise on machines with conventional mesh connected topologies. These algorithms use the coterie network to emulate wormhole routing [6,4] in that packets are not queued at intermediate nodes when progress is not possible, rather they are left in place on their current PE. One of the algorithms, at the cost of extra overhead, also emulates cut-through routing in that it sends packets not just to the next PE each time step, but rather to the next *free* PE, no matter how far away that may be. ROUTE uses the global feedback capability between controller and array to dynamically select between routing algorithms, thus presenting the programmer or language designer with a transparent mechanism for interprocessor communication.

2 The CAAPP

The CAAPP is the low-level processing array of the three-level architecture called the Image Understanding Architecture (IUA) being developed at the University of Massachusetts and Hughes Research Laboratories. The IUA is designed to perform real-time machine vision by combining pixel level, token level, and symbolic computation in one machine. The pixel level processor (the CAAPP) consists of a 512×512 content addressable array of one-bit processing elements (PEs). Each processing element has several general purpose registers, 320 bits of on-chip cache memory, 32K bits of main memory, and an "Activity Register" which is used for branching control. The PEs form a single instruction stream, multiple data stream (SIMD) array, with control provided by the Array Control Unit (ACU) which broadcasts instructions, data addresses, and global data. The controller

can also extract information from the array by associative polling, as hardware support is provided for Get-Some/None and Get-ResponderCount operations. The Get-Some/None operation is especially useful in determining whether a data dependent algorithm has completed, while Get-ResponderCount can be used for adaptive algorithms.

Communication between PEs themselves can take place in two different ways: by using the nearest neighbor mesh interconnection network, and via broadcast. In the second method, broadcasting PEs transmit information by writing to a specified register connected to the Some/None circuit. Receiving PEs then read a register which will have been set to the OR of the broadcast signals. Nearest-neighbor moves and broadcast are extremely efficient on the CAAPP: a 32-bit move takes around $3\mu\text{s}$ while broadcast from any set of PEs to the entire 512×512 array takes $35\mu\text{s}$.

One powerful addition that the CAAPP has over conventional associative processors is the coterie network, used to isolate the propagation of broadcast to specified regions. Each PE in the CAAPP controls a set of switches in four different directions (north, south, east, west) that enable the creation of electrically isolated groups of PEs sharing a local associative Some/None feedback circuit. These switches are set by loading the corresponding bits of the mesh control register in each PE with the appropriate mask. Because each PE views the mesh control register as local storage, coterie configurations can be loaded from masks stored in memory, or can be based on data dependent calculations. Isolated groups, or coterie, of processors can then respond to globally broadcast instructions in a locally data dependent fashion. For example, when a set of PEs executes a broadcast instruction, the receiving PEs will read the OR of precisely those PEs within the same coterie.

One way the switches of the coterie network can be set is so the columns or rows are isolated. Once this is done, the row or column "buses" can be arbitrarily segmented still further. The coterie network can thus emulate the mesh with reconfigurable buses [10], and the polymorphic-torus [8]. Another obvious use for the coterie network is in finding and labeling connected components. Each PE tests its four neighbors and compares the value with its own: if the values are equal, the PE closes its coterie switch in that direction; if not equal, the switch is opened. Using a standard leader election algorithm, a "master" PE from the newly formed component is selected. The leader broadcasts its unique ID to label the rest of the PEs in the component.

3 Routing on a Mesh

Much work has been done on the problem of routing on a mesh. Both SIMD and MIMD models have been used: In

the former no queues are used; in the latter queue size becomes a variable to be minimized. In the following discussion, square meshes are assumed and N refers to the total number of PEs, while $n = \sqrt{N}$ is the number of processors in a row or column. The lower bound on mesh routing is $2n - 2$ on the MIMD model and $4n - 4$ on the SIMD model; this is the minimum number of routing steps required for processors in opposite corners to exchange packets. The difference occurs because, in the MIMD model, different sets of processors can send packets in different directions on the same time step, while in the SIMD model, the direction must be the same for every packet. When there is wrap-around, the lower bounds are halved.

One way to route using the MIMD model is to use a simple greedy algorithm: First route each packet along the column to the correct row, then along the row to the correct column. Packets arriving at the correct rows are ordered in queues so that the ones that need to travel the furthest are given priority. This algorithm takes $2n - 2$ steps, but requires queues of size $\theta(n)$. A randomized routing algorithm due to [14] is an extension of greedy routing. The algorithm consists of three phases; randomize packets within the columns, send packets to correct column along the row, and send packets to correct row along the column. This algorithm will result in routing in $\approx 3n$ steps with a queue size of $O(\log N)$ with overwhelming probability. [7] has developed a more complex algorithm that is both optimal and uses constant size queues, although "it does not appear that the constant bound on the resulting queue size will be practical for moderate values of n (say, $n \leq 100$)."

Permutation routing can always be accomplished by sorting destination keys; currently this may be the best general-purpose on-line method for routing on the SIMD model. [13,11] use variations of bitonic sorting to sort a mesh into various standard orderings in $14n + o(n)$ routing steps, which turns out to be a factor of 4.5 from optimal for any n . When off-line calculation is allowed, better results can be achieved. [12] presents an algorithm, that with $O(\log^2 N)$ preprocessing time, can route optimally the class of permutations that can be specified by permuting and complementing the bits in the PE ID. [1] presents an algorithm that with preprocessing can route any permutation in $3n$ routing steps on a mesh with wrap-around.

To determine the applicability of these routing algorithms in terms of forming the basis for a general construct, we must first examine more closely exactly what we are trying to accomplish. The routing primitive should:

- require minimal preprocessing;
- be able to route permutations, as well as support intermediate combining of results in many-to-one

routing;

- route sparse as well as dense patterns efficiently;
- be able to take advantage of regularities, but not be too susceptible to congestion; and
- perform close to optimally.

The MIMD algorithms can be eliminated immediately because of their need for queues or heaps; simulating dynamic structures on a SIMD processor with no indirect addressing or index capability increases the routing complexity by a factor on the order of the maximum queue length or index offset. Since the MIMD algorithms all require queues of length greater than 4.5, the SIMD sorting methods would be preferable. But SIMD sorting and off-line routing also do not fulfill the need: they do not extend well to combining, nor do they take advantage of sparse routing. More comparisons will be made later, but first we will present a primitive that meets all of the criteria stated above.

4 The ROUTE Primitive

The ROUTE primitive consists of three similar, but distinct, algorithms which are selected dynamically through the use of the ACU get-count command. The basic idea of all three algorithms is to route greedily without the use of queues. Taking wormhole routing for inspiration [6,3], every PE simulates two channels, X and Y, that are chosen arbitrarily to run in directions parallel to the rows and columns respectively. We will first present three routing algorithms, then the method they are selected, followed by a description of how they have been modified to create a combine operation, and end the section with a randomized version of one of the algorithms.

The mesh greedy routing algorithm (MGRA) runs as follows: A PE uses the nearest neighbor connections to send a packet along the X-channel a distance of one PE per routing step, until the correct X coordinate (column) is reached. At this point the PE moves the packet from its X-channel to its Y-channel. The packet then continues along the Y-channel until the destination is reached. The X- and Y-moves are interleaved so that each occurs on every iteration of the algorithm. Packets travel in only one direction in each channel and wraparound is used, as in [4], so there will be no deadlock. If the packet has reached the correct X coordinate but the Y-channel at that PE is occupied, then the packet is "blocked", as are all the other packets contiguously behind that packet in the X channel. Y-channels are never blocked, so overall progress is assured. The critical question in running this algorithm without queues is how to inform those packets which are contiguously behind a blocked packet, that they too are blocked. In normal meshes, this notification step would require n steps, the maximum possible

number of blocked packets in a channel, and would yield a $\theta(N)$ algorithm. But by using the broadcast buses of the coterie network communication can be accomplished in $n/50$ machine cycles (the time it takes a broadcast signal to traverse the longest possible row bus), or on the order of a microsecond for a 512×512 array. The details of the notification step are as follows: all contiguous packets in X-channels form coterie or electrically isolated islands. The blocked packets open their left switches so that only packets behind them will receive the message, and then broadcast "blocked bits" to these coterie.

In the second algorithm, the four channel MGRA, two more simulated channels, X2 and Y2, are added to route packets in the opposite direction of X1 and Y1 respectively. The advantage is that packets can now be routed by a shortest path, minimizing the distance packets must travel. The number of iterations should be cut roughly in half. The disadvantage of the four channel MGRA is that the overhead is slightly more than doubled as there are now four ways that the X- and Y-channels can interact, instead of one. In the rest of the paper, the two channel version will be the default unless the number of channels is explicitly mentioned.

The third algorithm, the Coterie Greedy Routing Algorithm (CGRA), again uses two channels. The CGRA differs from the MGRAs in that the coterie network will be used to transfer packets, rather than just status bits. The key difference is that here, rather than moving packets just one PE at a time, all of the open space between occupied PEs is traversed in a single iteration of the algorithm. The mechanism is to create coterie which have the property that the rightmost PE (bottommost if these are Y-channels) contains a packet, while all other PEs in the coterie do not. The occupied PE then broadcasts its packet to the coterie, where it is read either by the destination or the furthest PE. Clearly the overhead per iteration is much higher for the CGRA than for the MGRA; exactly how much will be discussed below.

Selection among the three algorithms to create the unified ROUTE primitive is effected through use of the get-count directive of the ACU. The CGRA has much higher overhead than the MGRAs, but is very effective if the route is sparse and packets can be sent long distances during most iterations; therefore the CGRA will be called if the density of PEs sending packets is low. Similarly, the four-channel MGRA is used when the maximum distance that any packet must travel is somewhat less than the half the diameter of the torus. The density and the maximum distance can both be calculated in under 20 microseconds, or about one iteration of the MGRA. ROUTE is tuned so that the MGRA is the default; the more data dependent CGRA and four channel MGRA are only selected if it is virtually certain there will be a speed-up.

A combine operation has been created by augmenting the routing algorithms as follows: Instead of simply moving the packets that have arrived at their destinations from the Y-channel(s) to the output buffer, a binary operator is interposed. For example sum-combine adds the value in the packet to the value already in the output buffer. Many-to-one routing is implicit in the combine operation; more congestion is therefore likely to occur than in permutation routing. To deal with this situation, intermediate combining at the point of collision may optionally be executed. The cost is an increase in overhead of an extra compare and arithmetic operation for each cycle, but there are certainly situations where intermediate combining is worthwhile. One example is the degenerate case where the entire array is combined at one destination: the complexity of the combine operation is reduced from $O(N)$ to $O(n)$.

Some results from later in this paper are that the MGRA routes random permutations in slightly more than $2n$ iterations with very small standard deviation, while on non-trivial permutations arising from specific applications the number of iterations ranges from $2n$ to $3n$ iterations. It may be possible, however, that an application exists where the worst case takes longer than $3n$, and for which a highly predictable completion time is required. In this case we can take advantage of the small variance of the MGRA on random permutations by first randomizing the input packets along one dimension (as in [14]), and then routing to the destination. Since randomizing in one dimension takes n and random permutations take about $2n$ routing steps with very small variance, this algorithm routes all permutations in $3n$ data moves with extremely high probability.

5 Performance

In order to predict the performance of the MGRA and CGRA, variations of these algorithms were simulated at two levels of granularity. The first is a coarse simulation which disregards the number of time-steps needed to perform each iteration. The second is a complete assembly language implementation that was run on the IUA simulator [16].

5.1 Coarse Simulation

The coarse simulation was used to discover relationships among the following parameters: the algorithm used, the number of iterations needed for completion, the width of a side of the torus (n), the number of channels, the density of the route (that is, the percent of the N PEs which send a packet), and the number of times that individual packets were blocked. By disregarding the details of how the PEs actually carry out the bit-serial operations, it was possible to generate a larger number of trials than

would be possible using the full simulator, thereby reducing the standard deviation in the averages. Some of these experiments are now described.

- How many iterations do the various algorithms need to complete the route on random permutations, and how does this relate to the width of the torus?

The MGRA needed a number of iterations approximately equal to the diameter of the torus ($2n$); e.g. when $n = 64$ the average number of iterations was 134, when $n = 256$, the average number of iterations was 523. The four-channel MGRA needed slightly more than half that many iterations. In all cases, the standard deviation was less than 3. The performance of the two channel CGRA was sublinear: the relation $iters = n^{.89}$ fits the curve well, but no underlying structure was found.

- How much speed-up does each algorithm achieve when the number of packets to be routed in random permutations is reduced?

The performance of the MGRAs remains roughly the same as the density is decreased: there is still a high probability that some packet will need to travel close to the maximum distance. The CGRA achieves a significant speed-up because it takes advantage of the empty space between PEs. The number of iterations needed decreases very rapidly when the density is less than 20%; the break-even point between the CGRA and the MGRA occurs when the density is around 10%. For a 512×512 array, fewer than 60 iterations are required by the CGRA, whereas over 1000 are needed by the MGRA.

- How often is the most-blocked packet blocked during random permutations? What is the average number of times that a packet is blocked?

The average maximum number of times that any packet is blocked during the running of the MGRA is asymptotic to about 30. The average number of total blocks for all packets is roughly linear with the total number of packets; the average number of blocks per packet stays under 1 for the range tested, i.e. $n \leq 512$.

- How does the MGRA perform on particular permutations?

The MGRA was tested on numerous particular permutations and some basic results are presented above. The first table contains permutations of the type described in [12], that is, permutations "that can be specified by the permutation and complementing of bits in a PE address." The same notation is also used. The second table contains some permutations that cannot be thus specified.

Some PE ID "Bit" Permutations

Name	Formulation	Iters.
Bit Reverse	$[0,1,\dots,p-1]$	$< 2n$
Unshuffle	$[p-2,p-3,\dots,0,p-1]$	$< 2n$
Shuffle	$[0,p-1,\dots,1]$	$< 2n$
Transpose	$[p/2-1,\dots,0,p-1,\dots,p/2]$	$< 2n$
Shuffled Row-Major	$[p-1,p/2-1,\dots,p/2,0]$	$< 3n$
Bit Shuffle	$[p-1,p-3,\dots,1,p-2,\dots,0]$	$< 3n$
Vector Reverse	$[-(p-1),-(p-2),\dots,0]$	$< 2n$
Random Bit	[arbitrary orderings]	$< 3n$
with flipping	[and arbitrary flippings]	$< 3n$

Some Other Common Permutations

Name	Iters.
Random	$\approx 2n$
Reflection in X	$< 2n$
Reflection in Y	$< 2n$
Snakelike Row-Major Ordering	$\approx n$
Snakelike Column-Major Ordering	$< 2n$
$90^\circ * k$ rotation	$< 2n$
$45^\circ \pm 90^\circ * k$	$< 3n$
F-ordered vectors	$\approx 2n$

Trials were run for $n = 4, 8, 16, \dots, 256$ on all particular permutations. For the random permutations, from 100 ($n = 256$) permutations to many thousands were run for each n value.

Whenever the value in the "Iters" column states that less than $2n$ iterations are required to complete the MGRA, this indicates that no packets are blocked for any n tested. For values of $\approx 2n$ iterations, a small constant more than $2n$ was required, as with the random permutations mentioned above.

The 45° rotation route was calculated by using the standard rotation matrix and is a special case as it is not a permutation. The factors of 45° are the worst cases of all rotations tested ($0^\circ \dots 360^\circ$ in increments of 5°).

3.2 Implementation and Timing

Three variations of the greedy algorithm were implemented on the full CAAPP simulator [16], the MGRA, the 4-channel MGRA, and the CGRA. Some implementation details are worth mentioning before timing is discussed.

- The CAAPP is a bit-serial processor with a memory hierarchy: Each PE contains 320 bits of on-chip memory, any of which can be transferred to a nearest-neighbor PE in a single cycle, and 32K bits of off-chip memory. Therefore, execution times for the algorithms are linear in the number of bits in the message up to 120; for longer messages, the execution time per message length increases slightly.

- The execution time of the broadcast instruction is linear with respect to the diameter of the region in which the message is being sent. On the CAAPP, each bit propagates 50 PEs per instruction cycle. Therefore, while the broadcast instruction is technically $O(\sqrt{n})$, in practice it requires at most 6 cycles per bit on a 256×256 array. To simplify timing comparisons between simulations on different sized arrays, it is assumed that all broadcast instructions require the worst case of 6 cycles per bit, regardless of the diameter.

- Global feedback between ACU and array is used to determine whether the algorithms have completed and to increase performance. At the end of every iteration, the ACU executes get-SOME/NONE operations on the X- and Y-channels (each taking 2 microseconds); if the X-channels are empty, then they need no longer be simulated; if the Y-channels are also empty, then the algorithm has finished.

Presented are average times in milliseconds for random permutation routes.

2-channel MGRA

number of bits	width of torus	time
1	64	1.31
1	256	4.65
8	64	1.74
8	256	6.29
16	64	2.31
16	256	8.42

The 4-channel MGRA yielded results similar to the 2-channel MGRA, but has the advantage that the times are proportional to the maximum distance that any packet must travel, rather than to the diameter.

4-channel MGRA

number of bits	maximum distance	time
16	10	.50
16	15	.72

The CGRA is dependent both on the width of the torus and the density, so results are given with respect to both of these quantities. The last entry indicates that for very low densities, however, the expected time is roughly independent of the torus width.

CGRA

number of bits	width of torus	density	time
16	64	10%	1.95
16	64	4%	1.41
16	256	10%	4.83
16	256	4%	2.81
16	≤ 300	$\leq 5 * n$.74

We compare these results to a machine with a dedicated routing network: The 256×256 version of the Connection Machine II has running times of 500 micro-seconds for 32-bit permutations, and 80 micro-seconds for nearest neighbor permutations [9]. Therefore, the CAAPP running the greedy routing algorithms on random permutations has running times from roughly equivalent to an order of magnitude slower than the CM-2. When the CAAPP executes nearest neighbor permutations, the running time is less than 2 microseconds for 16 bits, or more than an order of magnitude faster than the CM-2.

6 Applications

In this section we will present some applications on the CAAPP for which interprocessor communication plays a significant role. This will give an idea of where the ROUTE primitive is useful, where the nearest neighbor moves are sufficient, and which algorithm of ROUTE was used. One result of this presentation is that the results from the previous section on random permutations extend to particular cases; in all applications tried so far, ROUTE terminates after at most $3n$ iterations.

Window operations. Local thinning and the convolutions used in differential edge detection are some of the many applications that use communication between PEs and a well specified neighborhood or 'window'. It is these sorts of operations that SMCCs perform extremely efficiently using nearest neighbor moves, and that is also the best way to execute window operations on the CAAPP. For example, a 3×3 Sobel operator can be executed, using a standard optimization, in 6 nearest neighbor moves, each taking only a few micro-seconds.

FFT. The FFT actually has two distinct sections, the initial phase where data is combined, and the final phase where the results are routed back, or unscrambled. The first phase requires very dense long-distance communication, as every PE in the array sends data 1, then 2, then 4, ... up to $n/2$ PEs away in both dimensions. This phase requires a total of $4n$ nearest neighbor moves. The unscramble phase uses the MGRA. Packets are never blocked for diameters tested ($n \leq 512$) so the number of iterations is always less than $2n$.

Some Matrix Operations. Matrix transpose, and reflection in the X and Y axes all used the MGRA part of ROUTE. As in the FFT unscramble, these permutations are non-blocking and so require less than $2n$ iterations.

Image Rotation. Although rotating an image is not trivial because of aliasing problems, effective methods can be constructed where most of the complexity consists of moving packets to locations generated by the rotation matrix. Rotations about the center were tested in increments of 5 degrees on several diameters and three

basic results obtained. First, for small rotations, the 4 channel MGRA was used and the number of iterations was small for a 64×64 mesh the number of iterations requires for 5, 10, 15, and 20 degrees is 10, 16, 20, and 27 respectively. Second, for rotations of 90, 180, and 270 degrees, the MGRA does not block, and therefore less than $2n$ iterations are required. Third, the worst case for the MGRA occurs for angles of 45, 135, 225, and 315, where up to $3n$ iterations are needed.

Ray Tracing. In lens design evaluation the problem arises of determining the distribution in the focal plane of rays passing through the lens from a point source. The PEs represents a point both on the lens and in the image plane. Each PE computes the path that a ray will take through "its" point on the lens and determines the address in the image plane where that ray will strike. The CAAPP then uses a sum-combine to route this information and obtain the result. In the case of an ideal lens, where all rays converge on one point, ROUTE takes $2n$ iterations with intermediate combining. In real designs, 140 to 150 iterations were needed on a 64×64 image.

Hough Transform. One algorithm for performing the Hough transform on the CAAPP runs as follows: The input consisting of an image plane (X,Y), usually a bit plane of thresholded edge pixels, and the Hough plane (ρ, θ) are both mapped to the rows and columns, respectively, of the mesh of PEs. For every θ from 0 to 180 degrees for I increments $d\theta$, $\cos(\theta_i)$ and $\sin(\theta_i)$ are broadcast by the controller. Each PE which is occupied in the (X,Y) plane must calculate the ρ value $\rho = X \cos(\theta_i) + Y \sin(\theta_i)$ and then send a bit to the PE at (ρ, θ_i) , which adds it to the total already there. The Hough transform can be viewed as a series of I column-histogram operations, one for every θ_i . Each column-histogram is executed in two steps; first a sum-combine operation is executed to get values from (X,Y) along the X axis to the correct ρ value, that is, to (ρ, Y) . Then, since all values in the various (ρ, Y) will be going to the same column, that is, the same θ , a row parallel prefix is executed in $\log n$ routing steps.

Region Characterization. (This example and those from the next few sections do not indicate the way that these algorithms are implemented on the IUA, but rather show how ROUTE could be used on a processor with reconfigurable buses but not coterries.) In order to decide whether to merge regions it is necessary to collect, in a "master" PE of that region, information such as number of points, number of border points, average and extremal spectral values, and other quantities. Characterization can be implemented using a simple leader election algorithm ($\log n$ steps) to select a region master, and the four channel MGRA with sum-combining. The number of iterations is very close to the maximum distance from any point of a region to its master PE as most collisions are handled with intermediate combining.

Region Merging. Assume a region merging algorithm, with preliminary regions already selected. Assume further that each region has elected a master PE which has gathered information about its own region and has determined the leaders of the neighboring regions. It is then necessary for the master PEs to communicate with all of the surrounding masters in order to determine whether or not to merge. If a few thousand initial regions are selected in a 256×256 image, then the density of communicating PEs will be small and the CGRA efficient. As merging progresses, the density will become smaller and smaller and only a few iterations of the CGRA will be needed for communication.

Local Histogramming. Obviously global histogramming can be implemented using sum-combine of the MGRA. But in one segmentation algorithm [2], local histogramming is used to extract information about subgrids of an image, typically 32×32 , and create regions on the basis of that information. Sum combine with the four channel MGRA can also be used for this procedure.

Convex Hull. On the CAAPP it is possible to find the convex hull of any number of sets of points simultaneously, as long as they are all members of the same region. A variation on the Graham scan [5] can be implemented which requires communication between a master PE and hull elements. In most cases, the density of hull points to total points is small, so that the CGRA will be efficient.

The applications just presented represent only a small fraction of the possible uses for the ROUTE primitive, but we emphasize three points. First, in dense, long distance, permutations the MGRA usually terminates in $2n$ iterations. Second, in no case did the MGRA take more than $3n$ iterations. And third, many applications have been found which use sparse or dense, nearby communication, and for which therefore the CGRA and four channel MGRA are preferable. Times for these algorithms are highly data dependent, but will always be less than those for the MGRA on the same application; often the times are substantially better.

7 Conclusions and Future Work

We have presented a routing primitive that is easy to use and that performs favorably to other available methods in many respects. One drawback is that these are not optimal algorithms: therefore, if maximum performance is required for a permutation known a priori, and for which an optimal route can be derived (e.g. bit permutations of PE IDs [12]), then naturally the optimal route should be implemented. However, ROUTE has also performed well on the bit permutations tested, as well as for many other applications including those requiring irregular, sparse, and many-to-one communication. Also, the randomized routing method (described at the end of section four) bounds the communication time to $3n$

MGRA iterations, although its use has not yet proved necessary. Perhaps just as important as performance, ROUTE gives the programmer and language designer a unified construct to handle all interprocessor communication that cannot easily be coded with nearest-neighbor moves.

A research problem that remains is a formal method for determining when to use nearest neighbor moves and when to use ROUTE. This issue is tied to a basic question of SIMD parallel processing, whether or not the mapping of data to PEs should be transparent to the programmer. In the primary CAAPP application of low-level machine vision, where the data often consist of pixels corresponding directly with PEs, we much prefer to know how our data are mapped. Corollaries of knowing the mapping are that nearest neighbor moves are easy to conceptualize and implement, and that is almost always obvious when the nearest neighbor connections and when general routing should be used. Needless to say, we view this flexibility as a great advantage.

One overall conclusion that can be drawn from this work is that the coterie network extends the applications the CAAPP can handle efficiently beyond SMCCs and into the range of SIMD machines with dedicated general routing networks. Obviously a dedicated routing network will better handle some classes of communication, the CM II routes random dense permutations more than ten times faster than ROUTE on the CAAPP does. But conversely, sparse and nearby permutations are handled with similar speed, and most importantly, no sacrifice is made in the areas where the CAAPP excels: nearest neighbor and broadcast communication. Therefore, in applications domains such as low-level machine vision where most of the inter-PE communication is sparse or involves neighborhoods, the cost of a dedicated routing network may not be justified if you have coterie.

Acknowledgments

We would like to thank Army Rosenberg, Jim Burrill, Mike Rudenko, and Mike Scudder for their helpful comments. This research has been supported by the Defense Advanced Research Projects Agency under contracts DACA76-89-C-0016, and DACA76-86-C-0015.

References

- [1] F. Annexstein and M. Baumslog (1990): "A Unified Approach to Offline Permutation Routing," *2nd ACM Symp. on Parallel Algorithms and Architectures*.

- [2] J.R. Beveridge, J. Griffith, R.R. Kohler, A.R. Hanson, E.M. Riseman (1989): "Segmenting Images Using Localized Histograms and Region Merging," *International Journal of Computer Vision*, 2.
- [3] W.J. Dally and C.L. Seitz (1986): "The Torus Routing Chip," *Distributed Computing*, 1 (3).
- [4] W.J. Dally and C.L. Seitz (1987): "Deadlock Free Routing in Multiprocessor Interconnection Networks," *IEEE Trans. on Comp.*, 36 (5).
- [5] R.L. Graham (1972): "An Efficient Algorithm for Determining the Convex Hull of a Planar Set," *Information Processing Letters*, 1.
- [6] M.C. Herbordt, C.C. Weems, D.B. Shu (1990): "Routing on the CAAPP," *Proceedings of the 10th Int. Conf. on Pattern Recognition*.
- [7] M.C. Herbordt, C.C. Weems, J.C. Corbett (1990): "Message Passing Algorithms on a SIMD Torus with Coteries," *Proceedings of the 2nd ACM Symposium on Parallel Algorithms and Architectures*.
- [8] P. Kermani and L. Kleinrock (1979): "Virtual Cut-Through: A New Computer Communication Switching Technique," *Comp. Networks*, 3.
- [9] F.T. Leighton, F. Makedon, I. Tollis (1989): "A $2n-2$ Step Algorithm for Routing in an $n \times n$ Array With Constant Size Queues," *1st ACM Symp. on Parallel Algorithms and Architectures*.
- [10] H. Li and M. Maresca (1987): "Polymorphic-Torus Network," *Proc. International Conference on Parallel Processing*.
- [11] J.J. Little, G.E. Blelloch, T.A. Cass (1989): "Algorithmic Techniques for Computer Vision on a Fine-Grained Parallel Machine," *IEEE Trans. on PAMI*, 11 (3).
- [12] R. Miller, V.K. Prasanna Kumar, D. Reisis, Q.F. Stout, (1988): "Meshes With Reconfigurable Buses," *Proc. of the MIT Conference on Advanced Research in VLSI*.
- [13] D. Nassimi and S. Sahni (1979): "Bitonic Sort on a Mesh-Connected Parallel Computer," *IEEE Trans. on Comp.*, 28.
- [14] D. Nassimi and S. Sahni (1980): "An Optimal Routing Algorithm for Mesh-Connected Parallel Computers," *J. of the ACM*, 27.
- [15] C.D. Thompson and H.T. Kung (1977): "Sorting on a Mesh Connected Computer," *Comm. of the ACM*, 20 (4).
- [16] L.G. Valiant and G.J. Brebner (1981): "Universal Schemes for Parallel Computation," *13th ACM Symp. on the Theory of Computing*.
- [17] C.C. Weems, S.P. Levitan, A.R. Hanson, E.M. Riseman, J.G. Nash, D.B. Shu (1989): "The Image Understanding Architecture," *International Journal of Computer Vision*, 2.
- [18] C.C. Weems and J.R. Burrill (1990): "The Image

Understanding Architecture and its Programming Environment," in *Parallel Architectures and Algorithms for Image Understanding*, V.K. Prasanna Kumar, ed. Academic Press, Orlando, Florida.

A Parallel Algorithm for List Ranking Image Curves in $O(\log N)$ Time*

Ling Tony Chen and Larry S. Davis

Computer Vision Laboratory, Center for Automation Research
University of Maryland, College Park, MD 20742-3411

Abstract

This paper describes an algorithm for ranking the pixels on a curve in $O(\log N)$ time using a CRCW PRAM model. The algorithm accomplishes this with N^2 processors for an $N \times N$ image. After applying such an algorithm to an image, we are able to move the pixels from a curve into processors having consecutive addresses. This is important on hypercube-connected machines like the Connection Machine because we can subsequently apply many algorithms to the curve (such as piecewise linear approximation algorithms) using powerful segmented scan operations (i.e. parallel prefix operations). The algorithm was implemented on the Connection Machine, and various performance tests were conducted.

1 Introduction

This paper considers problems associated with the efficient processing of image contours using hypercube connected massively parallel computers. While a significant amount of research has been devoted to the direct processing of images (e.g., convolutions, histogramming and more general parameter space clustering algorithms, stereo matching, time varying image analysis, etc.), comparatively little attention has been paid to the design and development of algorithms for processing contours. Although parallel algorithms have been developed for relatively simple tasks such as feature extraction (perimeter or area enclosed by a closed contour), there has been little practical experience with such algorithms and, furthermore, massively parallel algorithms have not been developed for more complicated tasks such as piecewise approximation or the determination of geometric relations between different contours extracted from an image or image sequence.

Contours might be marked in an image in a variety of ways. For example, one may apply an edge detection operator to an image, and then postprocess the binary edge map by a combination of thinning and segmentation at

vertices (i.e., edge pixels having more than two edge pixel neighbors) to obtain a set of simply connected contours (some open and some closed). Alternatively, an image may be segmented based on pixel spectral properties (the simplest example is segmentation by thresholding) and the boundaries of the components of the segmentation may be subsequently analyzed. In any event, we would like to design practical algorithms for computing representations for such contours that are useful for a variety of matching procedures (stereo, object recognition, etc.).

Our target machine for implementation is a 16K processor Connection Machine II [Hillis, 1985]. The Connection Machine is a hypercube connected massively parallel SIMD machine; an important advantage of the hypercube over the more common mesh network (e.g., MPP [Batcher, 1980], DAP [Reddaway, 1973]) is that one can efficiently compute parallel prefix, or scan, operations using the hypercube network. In Section 4, we describe scan operations and explain how they can be used to efficiently compute a piecewise approximation of a contour.

However, such scan operations can only be applied to a monotonic sequence of processor addresses. Since a contour can wind freely through an image, it is not generally the case that the sequence of processor addresses associated with the pixels on a contour will change monotonically as the contour is traversed. In Section 2 of this paper we present an efficient algorithm for ranking the pixels on a contour. Once the pixels are ranked, the contour can be moved to a new set of processors whose addresses will form a monotonic sequence (by simply moving the i th contour element to processor i). The algorithm that we present requires $\log N$ steps (where N is the length of the contour) on a CRCW PRAM. Note that unlike the list ranking algorithm originally presented in Wyllie [Wyllie, 1979], in which each element in the list to be ranked has only a single pointer, we must rank lists with two pointers (one to each neighbor on the contour except for the end points of open contours) and no preferred direction. (If the contours are detected by a sequential border following algorithm, or if the "inside" of a contour can be determined based on contrast, then a unique direction, say clockwise, can be assigned to contour elements. The algorithms described by Hung [Hung, 1988] and Wu [Wu *et al.*, 1989] depend on having such information available. Generally, however, this

*The support of the Defense Advanced Research Projects Agency (ARPA Order No. 6350) and the U.S. Army Engineer Topographic Laboratories under Contract DACA76-88-C-0008 is gratefully acknowledged.

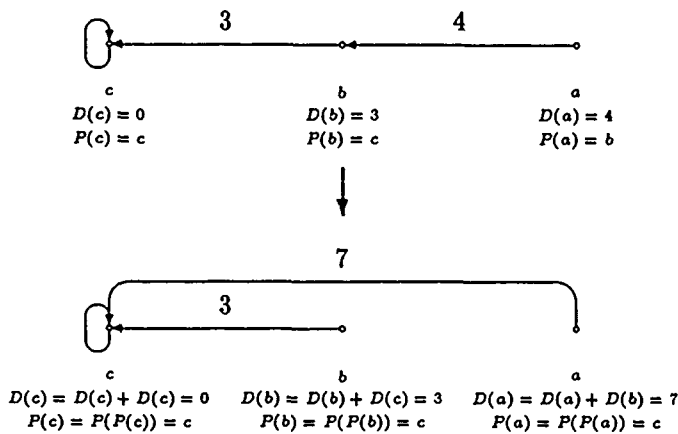


Figure 1: Distance doubling

is not the case, and some way must be found to break the symmetry without paying a severe computational penalty. The proof of the asymptotic properties of the algorithm is provided in Section 3. In Section 4 we give a few examples of using the scan operation on the new mapping. Section 5 discusses the actual implementation and compares the speed of the algorithm with respect to a linear time algorithm.

2 The Algorithm

In this section, an $O(\log N)$ CRCW PRAM algorithm for ranking pixels on a thinned curve is given. In Section 2.1 we discuss some basic terminology and operations. Section 2.2 contains an outline of the algorithm, while Section 2.3 describes the algorithm in detail. Finally, in Section 2.4 we describe the modifications needed to apply the algorithm to closed curves.

2.1 Basics

In the following discussion, we will use the term *pixel* and *pixel address* to represent the processor and processor address (whether virtual or physical) that holds that pixel. The algorithm utilizes two variables for each pixel. One is the variable P which is a pointer to another pixel on the curve, and the other, D , stores the distance between pixel i and pixel $P(i)$ along the curve. We shall use the notation $X(i)$ to represent the variable X in pixel i .

The algorithm can be applied to either 4- or 8-connected curves. Throughout the description of the algorithm we will discuss and display curves as if they were straight horizontal lines, in order to refer unambiguously to the left or right neighbors of a pixel. The image curves can be arbitrarily complicated, although they may not self-intersect.

In our algorithm we will frequently use the term *distance doubling* or simply *doubling*. Doubling is accomplished by performing $D(i) \leftarrow D(i) + D(P(i))$ followed by $P(i) \leftarrow P(P(i))$. The effects of a doubling operation are shown in Figure 1.

Throughout the algorithm we will also talk about *hooking* pixel i to pixel j . This involves simply changing

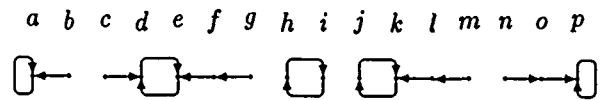


Figure 2: Example of ptr map after initial assignment

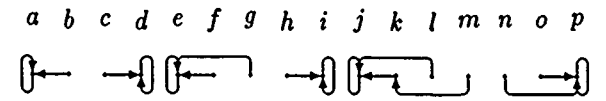


Figure 3: Figure 2 after doubling and adjustments

$P(i)$ to j and setting $D(i)$ to the distance between pixels i and j . There will always be a straightforward method of calculating this distance at the time of hooking.

2.2 Algorithm Outline

Initially, each curve pixel will store the addresses of the two pixels adjacent to it along the curve. Each pixel arbitrarily chooses one of its neighbors and places its address in P . We force endpoints to point to themselves, and we also force all pixels adjacent to endpoints to point to that endpoint. An example of what the pointers P on a curve might look like at this stage is shown in Figure 2.

We can now regard each curve as consisting of several segments by breaking the curve at gaps between pixels. Gaps occur between pixels where the pixel on the left of the gap is pointing left and the pixel on the right is pointing right.

In each segment (except for the segments containing endpoints) the leftmost pixel will always point right, and the rightmost pixel will always point left. In fact, examining the pixels from left to right, we encounter a single subsequence of pixels pointing right, followed by a single subsequence of pixels pointing left. A two-pixel loop occurs where these two subsequences meet.

Our goal is to convert each segment into either one or two segments, such that all pointers in a segment point in the same direction, and the last pixel along this direction forms a self-loop (i.e. points to itself). The two segments containing the endpoints are examples of segments that need no modification.

We accomplish this by first doubling each segment, followed by some local pointer adjustments. The result of applying the doubling and adjustments to Figure 2 is shown in Figure 3. The segment originally containing pixels c through g has been broken into two segments, one containing pixels c and d , and the other containing pixels e , f and g . All other segments in Figure 2 correspond to single segments in Figure 3.

Call the end of the segment with the self-loop the *head* of the segment and the other end the *tail*. Furthermore, call the pixel at the head of the segment the *head pixel*, and the pixel at the tail end the *tail pixel*. All segments with the head on the left are called *left segments*, and those with the head on the right are called *right segments*. So for example, starting from the left, the curve in Figure 3 would contain segments in the following order: left, right, right, left, right, left, right.

A segment is called *fully collapsed* (or simply *collapsed*)

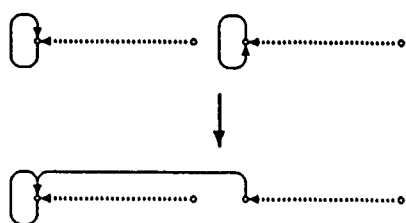


Figure 4: A head-tail merge

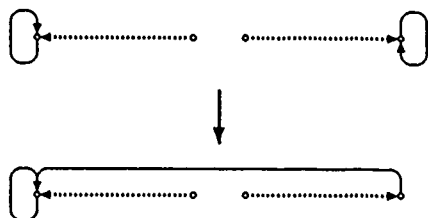


Figure 5: A tail-tail merge

if all the pixels in the segment point directly to its head pixel. A segment can be easily collapsed by repeatedly doubling it. Our goal is to merge the segments together, such that at the end of the algorithm we have only one segment. At that point, we simply collapse that segment by doubling.

Two types of merging are needed to guarantee that the segments are all merged in $O(\log N)$ time. The two types are *head-tail merge* and *tail-tail merge*. Head-tail merging, as shown in Figure 4, merges left segments with left segments and right segments with right segments. Figure 4 shows an example of two left segments being merged together. Tail-tail merging, (restricted to fully collapsed segments) on the other hand, merges left segments with right segments, provided that the left segment is to the left of the right segment. This is illustrated in Figure 5.

Basically, both types of merging simply *hook* the head of one segment to the head of the other segment. If this merging process is followed by a doubling, the two segments will become one and all pixels in the merged segment will point in the same direction.

2.3 Detailed Algorithm

The algorithm contains six steps, described below for pixel i . The code is executed in parallel by all active processors. The algorithm begins by initializing the variables P and D .

Step 1

$P(i) \leftarrow$ Maximum address of
the two neighbors of i
if (one of the neighbors of i is an endpoint)
 $P(i) \leftarrow$ address of that endpoint
 $D(i) \leftarrow 1$
if (i is an endpoint) $P(i) \leftarrow i; D(i) \leftarrow 0$

After Step 1, the pointer map will be similar to the one shown in Figure 2.

Step 2

Perform a distance doubling

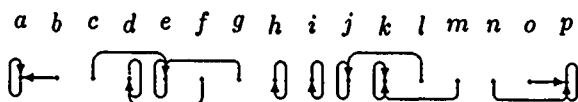


Figure 6: Figure 2 after doubling but before adjustments

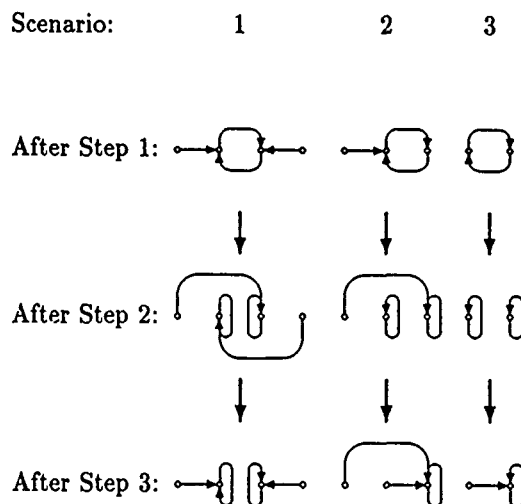


Figure 7: The scenarios of pointer adjustment

This initial doubling transforms all two-pixel loops (in the middle of the original segments) into a pair of self-looping pixels. The result of this doubling on the pointers in Figure 2 is shown in Figure 6.

This initial doubling can result in three types of degenerate segments, depending on whether there were pointers to the two-pixel loop of each segment from two, one or neither side of the loop. These three scenarios and the effects of the initial doubling on them is illustrated in Figure 7. In scenario 1, the doubling of Step 2 produces two segments that overlap with each other at the head pixels. In scenario 2, an extra self-loop occurs directly before the head pixel of the segment. In scenario 3, the result of doubling is two isolated self-loops. These two self-loops are unacceptable because for the merging process to work correctly each segment must have both distinct head and tail pixels.

The third step of the algorithm involves manipulating pointers of self-loops and their neighbors to realize the local structural transformations illustrated in Figure 7. Details of how these transforms are accomplished are described in Appendix A.

Step 3

Pointer adjustment to transform each curve into a sequence of non-overlapping, non-degenerate unidirectional segments.

Step 3 transforms each curve into a list of segments with each segment having a self-loop at one end, and all pixels in the segment pointing in the direction of the

self-loop. Steps 4 through 6 repeatedly apply merging and doubling to this list, finally transforming it into a single fully collapsed segment.

Note that after Step 3 not all segments are fully collapsed. Throughout the algorithm we will always have a combination of collapsed and non-collapsed segments. The non-collapsed segments benefit from the doubling in each iteration, while the collapsed segments benefit from the merging. A segment is guaranteed to be fully collapsed once its tail pixel points to its head pixel.

A tail-tail merge can only be performed on two fully collapsed segments. The result of this type of merge followed by a doubling results in a single collapsed segment that could be either a left or right segment. On the other hand, a head-tail merge can occur as long as the tail segment (the segment whose tail is adjacent to the head of the other) is fully collapsed. The head segment, on the other hand, need not be collapsed. Unlike the tail-tail merge, the head-tail merge could merge several segments simultaneously in one iteration. For example, a sequence of N collapsed left segments will be merged into a single non-collapsed left segment. While one might be tempted to collapse this segment at this point by repeatedly doubling $\log N$ times, this would lead to a worst-case $O(\log^2 N)$ algorithm. It is sufficient to perform a single doubling at this point. The alternation of merging and doubling guarantees that overall a sufficient number of doublings is performed to collapse the entire curve.

We next describe the tail-tail merge and head-tail merge in more detail. Step 4 performs a tail-tail merge, while Step 5 performs a head-tail merge. Finally, Step 6 is a single doubling. Since the tail-tail merge in Step 4 reverses the direction of one segment, the doubling algorithm in Step 6 must be suitably modified for this case. This is explained below.

repeat

Step 4 /* Tail-tail merge */

if ((i is the tail pixel of
a fully collapsed segment s_1) and
(one of i 's neighbor j is a tail pixel
of a second collapsed segment s_2) and
(the head pixel address of s_2 is larger
than the head pixel address s_1))
hook the head pixel of s_1

Step 5 /* Head-tail merge */

if ((i is the pixel address of
a head pixel of segment s_1) and
(s_1 did not participate in
the tail-tail merge of Step 4) and
(one of i 's neighbor is a tail pixel
of a different collapsed segment s_2))
hook i onto the head pixel of s_2

Step 6

Perform one doubling

until (no $P(i)$ changes in the iteration);

Note, that in Step 4, since s_1 is collapsed $P(i)$ points directly to the head of s_1 . Similarly, $P(j)$ points directly to the head of s_2 . It therefore requires no traversal of either s_1 or s_2 to hook the head of s_1 to the head of s_2 . Appendix A contains a more complete description of Steps 4 and 5. Also, note that during the tail-tail merge,

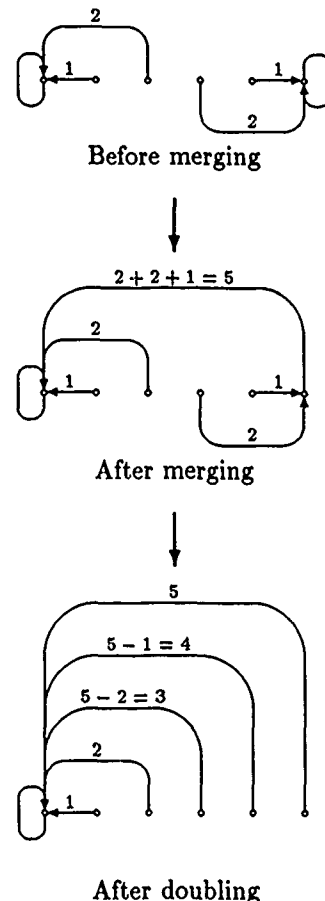


Figure 8: Tail-tail merge with distance doubling

the processor address is used to break the symmetry and decide which segment reverses its direction. The segment whose head has the smallest address reverses its direction and points to the head of the other segment. When performing this address comparison, there is one exception to consider. If the head of one of the segments (say s_2) is an endpoint, while the head of the other (say s_1) is not, we must reverse s_1 . This guarantees that throughout the algorithm the two endpoints will remain head pixels, and that each curve will always have a left segment as its leftmost segment, and a right segment as its rightmost segment. At the very last iteration the algorithm will perform a final tail-tail merge. At the beginning of this iteration, the curve will only contain one left segment (on the left) and one right segment (on the right). The merge will transform these two segments into either one left or one right segment, depending upon whether the left or right endpoint has the larger pixel address.

Another point to note is that throughout the algorithm we always modify the distance variable D whenever we modify the pointer variable P . During a tail-tail merge however, the direction of one of the segments is reversed and the distance doubling must be modified to yield the correct distance in the reversed segment. Specifically, if k is the head of s_1 (the reversed segment)

then we first set $D(k)$ to $l_1 + l_2 + 1$ (where l_i is the length of s_i) during the merge, and then the distance doubling involves simply computing $D(k) - D(r)$ for each pixel r in s_1 . This is illustrated in Figure 8.

One final point is that the repeat loop of Step 4, 5, and 6 terminates when all $P(i)$'s don't change during the iteration. Since this test must be carried out in constant time, a concurrent write is needed at this point. This is the only place in the algorithm that a concurrent write is needed.

2.4 Closed Curves

The algorithm described above must be modified to operate on closed curves. The modification is simple, and involves adding the following step between Steps 4 and 5.

Step 4.5 /* Extra head-tail merge */
 if ((i is the pixel address of
 a head pixel of segment s_1) and
 (s_1 did not participate in
 the tail-tail merge of Step 4) and
 (one of i 's neighbors is a tail pixel
 of a different collapsed segment s_2) and
 (the head pixel address of s_2
 is larger than i))
 hook i onto the head pixel of s_2

Basically, this extra step is just a head-tail merge in which hooking occurs only if s_2 's head pixel address is larger than i (i.e. we only hook smaller heads onto larger heads).

The difficulty with closed curves is that once a cycle is introduced into the curve, doubling will never resolve it, and the repeat loop of Steps 4 through 6 will never terminate. Note that in the initial pointer assignment of Step 1, we do not arbitrarily choose a neighbor. We choose the neighbor with maximum address. This guarantees that we don't create a cycle to begin with. Throughout the algorithm, no cycles can be introduced anywhere except for the head-tail merge in Step 5. Consider a closed curve containing two or more segments. If all the heads in the curve are pointing in the same direction, a head-tail merge would occur between every adjacent pair of segments, and a cycle will be introduced after the merge. By adding Step 4.5, the above condition will no longer occur. This is because in Step 4.5 at least one pair of segments in the ring (say s_1 and s_2) will be merged together, and hence in Step 5 the segment right before s_1 will no longer be able to hook onto s_1 because s_1 has merged with s_2 and is non-collapsed at this moment. The final result is that no cycle is introduced in either Steps 4.5 or 5, and hence no cycle will be introduced throughout the algorithm.

3 Complexity Analysis

The main result of this section is:

THEOREM 3.1 *For a curve containing N pixels, the number of iterations of Step 4, 5, and 6 needed by the algorithm will be at most $\lceil \log_4 N \rceil + 1$*

Since we process all curves in an image in parallel, the number of iterations needed to process an image will be proportional to the logarithm of the longest curve. For an $N \times N$ image the longest possible curve has $O(N^2)$ pixels; thus this theorem proves that our algorithm takes $O(\log N)$ time.

We must first establish some definitions and lemmas before we can prove the theorem.

DEFINITION: The length of a segment is the maximum number of pointer jumps required to get from any pixel in the segment to the head pixel.

LEMMA 3.1 *Distance doubling transforms a segment of length n to a segment of length $\lceil \frac{n}{2} \rceil$.*

PROOF: The proof is trivial; just separately consider the cases where the length of the segment is odd or even. \square

DEFINITION: A segment has the Tail Pixel Farthest (TPF) property (or is a TPF segment) if the number of jumps from the tail pixel to the head pixel is equal to the length of the segment.

It should be clear that a fully collapsed segment is a TPF segment, since the length of the segment is one, which is equal to the number of pointer jumps from the tail pixel to the head pixel.

LEMMA 3.2 *Throughout the repeat loop of Steps 4, 5, and 6, all segments are TPF segments at all times, except that segments involved in a tail-tail merge temporarily are not TPF between Steps 4 and 6 of the iteration during which they are merged.*

PROOF: The fact that the TPF property is initially true after Step 3 for all segments is trivial.

Assume that in Step 4 a tail-tail merge occurs between two segments s_1 and s_2 , and that the head of s_1 was modified to point to the head of s_2 ; call the merged segment s' . Now s' is not a TPF segment because the pixels in s_1 now need two pointer jumps (one to the head of s_1 , the second from the head of s_1 to the head of s_2) to reach the head of the combined segment s' , while the tail pixel of s' (originally the head pixel of s_1) needs only one jump to reach the head. Because all segments involved in a tail-tail merge do not participate in a head-tail merge, none of these segments are further modified in Step 5. Note that before Step 4, both s_1 and s_2 were TPF segments and hence both are of length one (since their tail pixels point to their head pixels), thus making the combined segment after Step 4 have length two. Hence after Step 6 the length of the combined segment must be one, which means that s' is a TPF segment.

In Step 5 (head-tail merge), we only hook the head of s_1 onto the head of s_2 if s_2 's tail pixel points to its head pixel. The only way that the resulting segment could fail to be a TPF segment after this merge is that the number of pointer jumps from a pixel in segment s_2 to the head of s_2 is greater than the length of s_1 plus 1. However, since both s_1 and s_2 were originally TPF segments, s_2 must

have been fully collapsed, thus making it impossible for the above case to occur.

It is clear that Step 6 (doubling) will not change a TPF segment into a non-TPF one; thus it should be clear that the TPF condition will be always true for all segments, except for the case of the tail-tail merge mentioned above. \square

DEFINITION: The length of a list of segments, or a complete curve, is the sum of the lengths of its constituent segments.

LEMMA 3.3 *When a head-tail merge merges a sequence of left (right) segments into one left (right) segment, the length of the new segment is the sum of the lengths of the individual segments.*

PROOF: If a head-tail merge of two segments s_1 and s_2 occurs, and the head of s_1 is hooked onto the head of s_2 , then because of the TPF requirements, s_2 must have been fully collapsed and hence had length 1. The length of the combined segment is now $\text{length}(s_1)$ plus one additional pointer jump from the head of s_1 to the head of s_2 , which is $\text{length}(s_1) + 1$. This is clearly equal to $\text{length}(s_1) + \text{length}(s_2)$. By induction, we can easily show that the Lemma holds when a string of head-tail merges occur. \square

LEMMA 3.4 *If the length of a sequence of adjacent left (right) segments is L before Step 5, and the rightmost (leftmost) segment of the sequence is not fully collapsed, then the length of the sequence after Step 6 is at most $\lfloor \frac{2}{3}L \rfloor$.*

PROOF: Because every segment of length 1 (i.e. a collapsed segment) has a neighboring segment to its right (left), they will be merged into larger segments in Step 5. Because of this, and Lemma 3.3, all segments in the sequence will have length at least 2 after Step 5. After the doubling of Step 6, each segment of length n will be shortened into a segment of length $\lfloor \frac{n}{2} \rfloor$. Since $n \geq 2$, the worst case occurs when $n = 3$ and the resulting segment has length 2. If all the segments after Step 5 were of length 3, and the total number of segments were m , then $L = 3m$ and the final length of the sequence would be $2m$, which is equal to $\lfloor \frac{2}{3}L \rfloor$. It is tedious but easy to prove that for all other cases, $\lfloor \frac{2}{3}L \rfloor$ is an upper bound on the final length of the sequence. \square

LEMMA 3.5 *If the length of a sequence of adjacent left (right) segments is L before Step 5, and the rightmost (leftmost) segment of the sequence is fully collapsed, then the length of the sequence after Step 6 is at most $\lfloor \frac{2}{3}(L - 1) \rfloor + 1$.*

PROOF: Because we no longer can assume that all segments of length 1 will have an adjacent segment to the right (left) (because the rightmost (leftmost) segment now has length 1), we cannot use the previous proof. However, if the two rightmost (leftmost) segments are

both fully collapsed, these two segments will be merged in Step 5, and the property that all segments after Step 5 have length at least 2 will hold. The proof from the previous lemma can then be employed, thus making the upper limit on the final length $\lfloor \frac{2}{3}L \rfloor$. If on the other hand, the rightmost (leftmost) segment is fully collapsed, but the second from the right (left) is not, then we can apply the previous lemma to all the segments in the sequence except for the rightmost one, thus making the upper bound on the final length $\lfloor \frac{2}{3}(L - 1) \rfloor + 1$. Since $\lfloor \frac{2}{3}(L - 1) \rfloor + 1$ is always larger than $\lfloor \frac{2}{3}L \rfloor$, it is the final upper bound. \square

If we break each list of segments into several sequences at points where two heads meet (i.e. where a right segment meets a left segment and the left segment is on the right), each sequence, starting from left to right, will contain a series of left segments followed by a series of right segments.

DEFINITION: A maximal sequence (from left to right) of left segments followed by a maximal sequence of right segments is called a *left-right sequence*.

LEMMA 3.6 *If the length of a left-right sequence is L before Step 4, then its length after Step 6 is at most $\lfloor \frac{3}{4}L \rfloor$.*

PROOF: First consider the case in which no tail-tail merge occurred in Step 4 for this left-right sequence. This could happen if either the rightmost left segment or the leftmost right segment is not fully collapsed. Without loss of generality, let us assume that the rightmost left segment is not fully collapsed. Let the length of the sequence of left segments be l , and the length of the sequence of right segments be r , thus making $L = l + r$. By applying Lemma 3.4 to the sequence of left segments and Lemma 3.5 to the sequence of right segments, the upper bound on the length of this left-right sequence after Step 6 is $\lfloor \frac{2}{3}l \rfloor + (\lfloor \frac{2}{3}(r - 1) \rfloor + 1)$. By comparing $\lfloor \frac{2}{3}l \rfloor + (\lfloor \frac{2}{3}(r - 1) \rfloor + 1)$ with $l + r$, we can see that the highest ratio is $\frac{3}{4}$, and it occurs when $l = 3$ and $r = 1$. Furthermore, we can check that $\lfloor \frac{3}{4}L \rfloor$ is the upper bound for all cases of l and r by considering the nine cases where l is $3m, 3m + 1$, and $3m + 2$, and r is $3n, 3n + 1$, and $3n + 2$.

Now consider the case in which a tail-tail merge did occur between the rightmost left segment and the leftmost right segment. Because of the TPF property, these two segments were originally fully collapsed, and hence after Step 6 the final combined segment will be of length 1. Excluding the two segments involved in the tail-tail merge, assume the length of all the other left segments is l , and the length of all the other right segments is r , hence making $L = l + 2 + r$. By applying Lemma 3.5 on both sequences of left and right segments, the upper bound on the length of this left-right sequence after Step 6 is $(\lfloor \frac{2}{3}(l - 1) \rfloor + 1) + 1 + (\lfloor \frac{2}{3}(r - 1) \rfloor + 1)$. By comparing $(\lfloor \frac{2}{3}(l - 1) \rfloor + 1) + 1 + (\lfloor \frac{2}{3}(r - 1) \rfloor + 1)$ with $l + 2 + r$, we can see that the highest ratio is also $\frac{3}{4}$, and it occurs when $l = 1$ and $r = 1$. Furthermore, by checking

the same nine cases we can again see that $\lfloor \frac{3}{4}L \rfloor$ is also the upper bound for all cases of l and r . \square

LEMMA 3.7 *If the length of a curve is L before Step 4, then its length after Step 6 is at most $\lfloor \frac{3}{4}L \rfloor$.*

PROOF: Since each curve is just a list of left-right sequences, Lemma 3.6 immediately implies this Lemma. \square

The proof of Theorem 3.1 is quite straightforward at this point:

THEOREM 3.1 *For a curve containing N pixels, the number of iterations of Step 4, 5, and 6 needed by the algorithm will be at most $\lfloor \log_3 N \rfloor + 1$*

PROOF: A curve with N pixels will result in a list of segments with length less than or equal to N after Steps 1, 2, and 3. By Lemma 3.7, after k iterations of Step 4, 5, and 6, a curve of original length N should have length at most $(\frac{3}{4})^k N$. Hence after at most $\lfloor \log_3 N \rfloor$ iterations the length of the curve will be reduced to 1 (which means that the entire list has been fully merged and collapsed). One more iteration in which nothing is changed is needed so that the repeat loop terminates. Hence the maximal number of iterations needed is $\lfloor \log_3 N \rfloor + 1$. \square

4 Advantages of Remapping

Once a curve has been list ranked, we can efficiently compute many properties of the curve. The most simple example is the length of the curve. Prior to list ranking, it is not clear that the length can be determined in logarithmic time. A curve's length is a byproduct of our list ranking algorithm, because this length is simply the value of $D+1$ in the curve's tail pixel at the termination of the algorithm. A pixel can simply test if it is a tail pixel by checking that it is an endpoint and that it is not a self-loop (tests on closed curves are more complicated).

However, the principal advantage of the list ranking algorithm is that we can now pack the curve pixels into a stream of monotone contiguous processors. This packing can be easily achieved simultaneously by all image curves as follows:

1. Each tail pixel writes $D+1$ into the head pixel of its curve. The tail pixel locates this head pixel by pointer P . Assume that this value $D+1$ (which is the length of the curve) is stored in the variable L in the head pixel.
2. The head pixels of all the curves participate in a scan operation that calculates the prefix sum of the variable L . An *exclusive* scan operation is performed such that the sum in head pixel i does not include $L(i)$ itself. Assume the prefix sum is stored in variable S .
3. All pixels now perform a concurrent read from their head pixels, to obtain the value of S , and add it to their own local D value. Assume the result is stored in I .

4. I now contains a unique index number for each active pixel. Now all pixels can write information about themselves (such as x, y coordinates) to a unique processor, and all pixels of the same curve are guaranteed to be stored in monotone contiguous processors.

After packing all curve pixels into contiguous processors, we can more easily manipulate them by using segmented-scan operations. In Section 4.1 we briefly describe the scan and segmented-scan operations, while in Section 4.2 and 4.3 we show two examples of operating on curves using scan operations.

4.1 The Scan Operation

The scan operation [Kruskal *et al.*, 1985, Ladner and Fischer, 1980, Little *et al.*, 1989, Bletloch, 1988] takes a binary associative operator \oplus , and a list $[a_0, a_1, \dots, a_{n-1}]$ of n elements, and returns the list $[a_0, (a_0 \oplus a_1), \dots, (a_0 \oplus a_1 \oplus \dots \oplus a_{n-1})]$. Examples of binary associative operators frequently used are $+$, maximum, minimum, and, or, and first (first returns the first of its two arguments). We will henceforth call these scan operations $+$ -scan, max-scan, min-scan, and-scan, or-scan, and first-scan. Some examples are shown below:

A	$=$	[4	3	6	2	6	8	1	9]
$+$ -scan(A)	$=$	[4	7	13	15	21	29	30	39]
max-scan(A)	$=$	[4	4	6	6	6	8	8	9]
first-scan(A)	$=$	[4	4	4	4	4	4	4	4]

The above illustrations are inclusive scans. For each type of scan, there will also be exclusive scans, where the element at position i is not included in the scan result at position i . And there will also be reversed versions of all the scan operations, where the scan starts from the end of the list. Below are some examples of $+$ -scan:

A	$=$	[4	3	6	2	6	8	1	9]
inc $+$ -scan(A)	$=$	[4	7	13	15	21	29	30	39]
exc $+$ -scan(A)	$=$	[0	4	7	13	15	21	29	30]
inc, rev $+$ s(A)	$=$	[39	35	32	26	24	18	10	9]
exc, rev $+$ s(A)	$=$	[35	32	26	24	18	10	9	0]

Finally there are segmented versions of all the above scans. In segmented scans, the list is partitioned into several segments based on a segment-flag. In forward scans, a **True** segment-flag marks the beginning of a new segment, while in a reverse scan, a **True** segment-flag marks the end of segment. Examples of segmented scans are shown below:

Segment-flags	$=$	[T	F	T	F	F	T	F]				
A	$=$	[4	3	6	2	8	1	9]				
inc $+$ -scan(A)	$=$	[4	7]	[6	8	16]	[1	10]
exc $+$ -scan(A)	$=$	[0	4]	[0	6	8]	[0	1]
inc, rev $+$ s(A)	$=$	[7	3]	[16	10	8]	[10	9]
exc, rev $+$ s(A)	$=$	[3	0]	[10	8	0]	[9	0]

Scan operations are efficiently implemented on hypercube connected machines. By letting the head of each transformed curve set its Segment-flag to **True**, we can use segmented scan operations to efficiently compute many useful properties and representations of the curves. We provide two examples in the following subsections.

4.2 Calculating Least Square Fit Lines

Calculating a least square fit line for curves is a very common task, and can be very easily carried

out in our new mapping. For a sequence of points $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$, the least square fit line is simply

$$\begin{aligned} & \left(\sum_{i=0}^n y_i \sum_{i=0}^n x_i y_i - \sum_{i=0}^n x_i \sum_{i=0}^n y_i^2 \right) x + \\ & \left(\sum_{i=0}^n x_i \sum_{i=0}^n x_i y_i - \sum_{i=0}^n y_i \sum_{i=0}^n x_i^2 \right) y + \\ & \left(\sum_{i=0}^n x_i^2 \sum_{i=0}^n y_i^2 - \left(\sum_{i=0}^n x_i y_i \right)^2 \right) = 0 \end{aligned}$$

The coefficients of the line can be easily computed by the first processor of each segment once this processor has collected the values of $\sum_{i=0}^n x_i$, $\sum_{i=0}^n y_i$, $\sum_{i=0}^n x_i^2$, $\sum_{i=0}^n y_i^2$, and $\sum_{i=0}^n x_i y_i$. These values can be easily obtained by doing a reverse inclusive +scan on the values x , y , x^2 , y^2 , and xy respectively.

4.3 Piecewise Linear Approximations of Curves

Ramer [Ramer, 1975] devised a method of finding piecewise linear approximations of curves, by breaking curves at points that are farthest away from the line that connects the two endpoints of the curve. By repeatedly applying this curve breaking method until all pixels of each curve are within a threshold distance from the line connecting the endpoints, we can obtain a good piecewise linear approximation. Many more piecewise approximation algorithms are described by Pavlidis [Pavlidis, 1977].

Ramer's algorithm can be easily implemented on our monotone contiguous mapping between pixels and processors. The algorithm involves the following steps:

1. Perform a reverse first-scan on the x and y coordinates, so that the first processor of each curve segment has the x , y coordinates of the two endpoints of the segment.
2. The first processor of each curve calculates the coefficients of the line that passes through its endpoints.
3. A forward first-scan is performed to broadcast the coefficients of this line to all pixels in this curve.
4. All pixels calculate the distance between themselves and the line joining the endpoints.
5. A reverse max-scan is performed on this distance concatenated with the processor ID number of each pixel. After this is done, the first processor in each segment will know the processor ID number of the largest address processor having maximal distance from the line. Let m_i be the address of the processor having maximal distance in curve segment i .
6. If this maximum distance is smaller than a threshold, the segment *de-selects* itself and is idle through Steps 7 and 8. If all segments in the image *de-select*, the algorithm terminates.
7. A forward first-scan is used to broadcast m_i to all processors in curve segment i .

Curve Length	$O(\log N)$ alg		$O(N)$ alg	
	t(ms)	iter	t(ms)	iter
64	193	5	605	64
128	201	5	1219	128
256	244	6	2430	256
512	262	6	4859	512
1024	334	7	9696	1024
2048	487	8	19410	2048
4096	715	8	38827	4096

Table 1: Our algorithm versus linear time algorithm

8. Processor m_i sets its segment-flag to **True**, thus splitting curve segment i for the next iteration.
9. Steps 1 through 9 are repeated.

5 Implementation and Results

All of the above algorithms were implemented on the Connection Machine [Hillis, 1985]. The CM is a hypercube connected massively parallel SIMD computer, with between 16k and 64k 1-bit processors. Our system has 16k processors. Each processor has 64k bits of local memory. The CM system provides two forms of communication between processing units. The *router*, which provides the more general communication mechanism, allows any processor to communicate with any other processor, through the hypercube network, making the CM simulate a CRCW PRAM machine. The NEWS grid provides 2-D mesh communication, in which each processor can communicate with its 4-connected neighbors. NEWS communication is much faster than the general router.

Instead of forcing data to fit the machine size, the CM software supports the definition of *virtual processors*. For example, when processing a 64k pixel image, with only 16k processors on the CM, one can regard the machine as containing 64k virtual processors, with each physical processor simulating the work of four virtual processors. The term *VP ratio* denotes the ratio of the number of virtual processors to the number of physical processors.

Table 1 lists the results obtained by running the $O(\log N)$ list ranking algorithm on the CM for different curve lengths. The VP ratio was 8 for all these tests. As a comparison, we also implemented a linear time algorithm that simply propagates the ranking information along the curve using only the NEWS grid. The list ranking algorithm is much faster than the linear time algorithm, though each iteration is actually much slower (due to using the general router for communication). One can also observe the logarithmic nature of the algorithm by examining how the number of iterations changes as a function of the curve length.

There are a few modifications that can be employed to improve the running time of the algorithm. One is to perform more than one doubling in Step 6. There is a tradeoff here: performing more doublings in Step 6 will cause each iteration to run longer, but the number of iterations will be decreased. For different images on differ-

Pixels	Ramer's algorithm			
	C_{start}	C_{end}	iter	t(ms)
7706	581	800	5	36
6735	515	700	5	37
7943	630	877	6	46

Table 2: Result of applying Ramer's algorithms

Total PE & pixels		LSF algorithm	
PE	pixels	curves	t(ms)
8192	7706	800	30
8192	6735	700	30
8192	7943	877	30

Table 3: Result of applying LSF algorithm

ent size machines, the optimal number will vary, but our experiments indicate it is generally around two or three doublings. The times and iterations in Table 1 were generated using two doublings in Step 6. Had we used one doubling, the algorithm would have taken roughly 50% more iterations. Another way to improve running time is to perform a few doublings right after Step 3, but before the repeat loop of Steps 4 through 6. This usually decreases the number of iterations by one or two. We did not do this for our test in Table 1.

We also applied Ramer's piecewise linear approximation algorithm and the least square fit algorithm to three different 512×512 test images using the scan instructions available on the CM. The results are tabulated in Table 2 and Table 3. Edge detection and the list ranking algorithm were applied to three images and the remaining pixels were packed into monotone contiguous processors. At this point each image contained C_{start} curves, and after applying Ramer's algorithm contained C_{end} curves. A least square fit algorithm was then applied to these C_{end} curves to estimate their positions. Since 8K physical processors were used, the VP ratio was one for all three experiments because the number of edge pixels left was always less than the number of processors we had. Note that the least square fit algorithm takes roughly the same amount of time as four iterations of Ramer's algorithm. This is because the least square fit algorithm involves floating point calculations, whereas Ramer's algorithm does not.

6 Conclusion

We have described an algorithm that can produce a list ranking of pixels on a thinned curve. The algorithm runs in $O(\log N)$ time on a CRCW PRAM model of computation. This list ranking makes it possible to move the thinned edge pixels of an image into monotone contiguous processors, such that further processing of curves can be done both more easily and more efficiently.

References

[Batcher, 1980] Kenneth E. Batcher. The design of a

massively parallel processor. *IEEE Transactions on Computers*, 29:836-840, 1980.

[Blelloch, 1988] Guy E. Blelloch. *Scan Primitives and Parallel Vector Models*. PhD thesis, Massachusetts Institute of Technology, November 1988.

[Hillis, 1985] W. Daniel Hillis. *The Connection Machine*. MIT Press, Cambridge, MA, 1985.

[Hung, 1988] Yubin Hung. *Parallel Processing of Geometric Representations on SIMD Computers*. PhD thesis, University Of Maryland, College Park, 1988.

[Kruskal et al., 1985] Clyde P. Kruskal, Larry Rudolph, and Marc Snir. The power of parallel prefix. *IEEE Transactions on Computers*, 34:965-968, 1985.

[Ladner and Fischer, 1980] R. E. Ladner and M. J. Fischer. Parallel prefix computation. *JACM*, 27:831-838, 1980.

[Little et al., 1989] James J. Little, Guy E. Blelloch, and Todd A. Cass. Algorithmic techniques for computer vision on a fine-grained parallel machine. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:244-256, 1989.

[Pavlidis, 1977] T. Pavlidis. *Structural Pattern Recognition*. Springer-Verlag, New York, 1977.

[Ramer, 1975] U. Ramer. Extraction of line structures from photographs of curved objects. *Computer Graphics and Image Processing*, 4:81-103, 1975.

[Reddaway, 1973] S. F. Reddaway. DAP—a distributed array processor. In *First Annual Symposium on Computer Architecture*, pages 61-65, 1973.

[Wu et al., 1989] Angela Y. Wu, S. K. Bhaskar, and Azriel Rosenfeld. Parallel processing of region boundaries. *Pattern Recognition*, 22:165-172, 1989.

[Wyllie, 1979] J. C. Wyllie. *The Complexity of Parallel Computation*. PhD thesis, Cornell University, 1979.

Appendix A: Detailed Algorithm Code

The following is a detailed listing of the code for our list ranking algorithm. In our algorithm, we often need to compare pixel addresses in order to break the symmetry between pixels. In all these comparisons we always want the address of an endpoint to be larger than the address of a pixel in the middle of the curve. We accomplish this by augmenting each pixel address with an extra bit at the most significant bit position. This extra bit is 1 if the pixel is an endpoint, and 0 otherwise.

In the following code, i is used to represent both the pixel i and the processor address of the processor holding pixel i . The code is executed in parallel by all pixels i on all curves.

```

01  $P(i) \leftarrow$  Maximum of all  $j$ 
   such that  $j$  is a neighbor of  $i$ 
02  $D(i) \leftarrow 1$ 
03 if (endpoint( $i$ ))
   begin  $P(i) \leftarrow i$  ;  $D(i) \leftarrow 0$  end

```

Note that because we are using augmented addresses, $P(i)$ is guaranteed to point to an endpoint if i is right beside an endpoint. These 3 lines are equivalent to Step 1.

The next operation to perform is to double the pointers and adjust the pointers such that the pointer map looks like Fig. 3.

```

04  $OLDP(i) \leftarrow P(i)$ 
05  $P(i) \leftarrow OLD(P(i))$ 
06  $D(i) \leftarrow D(i) + D(OLDP(i))$ 
07  $LOOP(i) \leftarrow \text{false}$ 
08 if  $(i = P(i))$   $LOOP(i) \leftarrow \text{true}$ 
09  $POINTEDTO(i) \leftarrow \text{false}$ 
10 if (not  $LOOP(i)$ )
     $POINTEDTO(OLDP(i)) \leftarrow \text{true}$ 
11 if ( $LOOP(i)$  and
    (not  $POINTEDTO(OLDP(i))$ ) and
12 ( $POINTEDTO(i)$  or
    (not  $POINTEDTO(i)$ ) and
    ( $i < OLDP(i)$ ))) begin
13      $P(i) \leftarrow OLDP(i)$ 
14      $D(i) \leftarrow 1$ 
15      $LOOP(i) \leftarrow \text{false}$ 
16 end
17 if ((not  $LOOP(i)$ ) and  $LOOP(OLDP(i))$ )
    begin
18      $P(i) \leftarrow OLDP(i)$ 
19      $D(i) \leftarrow 1$ 
20 end

```

Lines 4 through 6 correspond to Step 2 and do an initial doubling. Lines 7 through 20 accomplish the pointer adjustments in Step 3. Lines 7 and 8 set up the Boolean variable $LOOP$ such that $LOOP(i)$ is true if and only if $P(i)$ points to i itself. $LOOP$ will be updated throughout the algorithm such that it always reflects whether $P(i)$ points to itself. Lines 9 and 10 set up the Boolean variable $POINTEDTO$ such that $POINTEDTO(i)$ is true if and only if there is some other non-self-looping pixel j that originally pointed to i before the doubling. The pointer adjustments for the three scenarios are done in two steps. First, lines 11 through 16 adjust self-loop pointers to point at the correct pixels, then lines 17 through 20 adjust the pointers that jump over loops. A look at Fig. 7 will help you understand how this code works. Lines 11 and 12 perform a complicated test to determine if the conditions of scenario 2 and 3 have occurred. If so, lines 13, 14, and 15 adjust the pointers such that the self-loop that needs adjustment is rehooked to the self-loop right beside it. Line 17 tests for the case of a pointer jumping over a self-loop (as in scenario 1) and lines 18 and 19 rehook this pointer to point to the self-loop, instead of jumping over it.

What follows is the code that performs the repeated iterations of merging and doubling.

```

21  $LOOP(i) \leftarrow \text{false}$ 
22 if  $(i = P(i))$   $LOOP(i) \leftarrow \text{true}$ 
23 repeat
24      $SAVEP(i) \leftarrow P(i)$ 
25      $N(i) \leftarrow \text{Address of neighbor } j$ 
        such that  $P(j) \neq P(i)$ , if no
26     such neighbor exists, let  $N(i) \leftarrow i$ 
27      $PN(i) \leftarrow P(N(i))$ 
28      $BUSY(i) \leftarrow \text{false}$ 
29      $REVERSE(i) \leftarrow \text{false}$ 

```

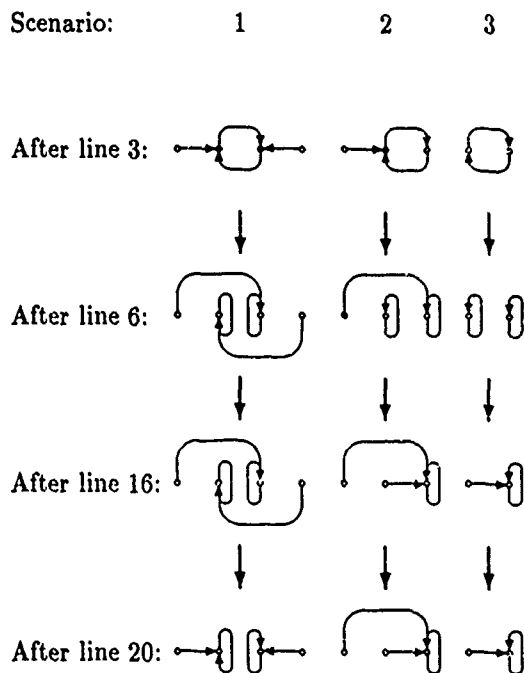


Figure 9: The two steps of pointer adjustment

```

30 if (( $PN(i) \neq P(i)$ ) and
    (not  $LOOP(N(i))$ ) and
     $LOOP(PN(i))$ ) begin
31     if ((not  $LOOP(i)$ ) and  $LOOP(P(i))$ ) begin
32          $P(P(i)) \leftarrow PN(i)$ 
33          $D(P(i)) \leftarrow D(i) + D(N(i)) + 1$ 
34          $REVERSE(P(i)) \leftarrow \text{true}$ 
35          $BUSY(PN(i)) \leftarrow \text{true}$ 
36          $LOOP(P(i)) \leftarrow \text{false}$ 
37     end
38     if ( $LOOP(i)$  and (not  $BUSY(i)$ ) and
        ( $i < PN(i)$ )) begin
39          $P(i) \leftarrow PN(i)$ 
40          $D(i) \leftarrow D(N(i)) + 1$ 
41          $LOOP(i) \leftarrow \text{false}$ 
42     end
43     if ( $LOOP(i)$  and (not  $BUSY(i)$ ) and
         $LOOP(PN(i))$ ) begin
44          $P(i) \leftarrow PN(i)$ 
45          $D(i) \leftarrow D(N(i)) + 1$ 
46          $LOOP(i) \leftarrow \text{false}$ 
47     end
48 end
49  $OLDP(i) \leftarrow P(i)$ 
50  $P(i) \leftarrow OLD(P(i))$ 
51 if ( $REVERSE(OLDP(i))$ )
     $D(i) \leftarrow D(OLDP(i)) - D(i)$ 
52 else  $D(i) \leftarrow D(i) + D(OLDP(i))$ 
53 until ( $\forall i$  we have ( $SAVEP(i) = P(i)$ ))

```

Line 30 tests for the conditions common to both the tail-tail merge and head-tail merge. Lines 31 through 37 do the tail-tail merge that corresponds to Step 4. Lines 38 through 42 do the first head-tail merge corresponding

to Step 4.5. Lines 43 through 47 do the second head-tail merge corresponding to Step 5. If no closed curves are in the image, we can omit lines 38 through 42, and the if statement in line 43 can be changed to just checking if ($LOOP(i)$ and (not $BUSY(i)$)) is true, since the condition of $LOOP(PN(i))$ has already been checked in line 30. The reason we rechecked whether $LOOP(PN(i))$ is true or not is because Step 4.5 (in particular line 41) might have changed this condition. Note how the variable $BUSY$ is used to mark the head of a segment participating in a tail-tail merge, so that the head-tail merges of the same iteration do not merge these segments. Finally, in lines 49 through 52 we perform the distance doubling for each iteration. Note how the Boolean variable $REVERSE$ is used as a flag to mark the presence of a tail-tail merge (in line 34) so that the distance D can be computed correctly in lines 51 and 52.

Purposive and Qualitative Active Vision

John (Yiannis) Aloimonos

Computer Vision Laboratory, Center for Automation Research,
University of Maryland, College Park, MD 20742-3411

ABSTRACT

A fundamentally new way of examining problems of visual perception is described in this paper. Up to now, vision was regarded as a recovery problem, i.e., as the problem of reconstructing an accurate representation of the 3-D scene and its properties from image cues such as shading, contours, motion, stereo, color, etc. This approach has contributed many theoretical results and has led to new mathematical techniques, e.g. related to regularization and discontinuities. But what is vision for? Why do animals have it and why do we want to understand it? The answer is, of course, that we need vision in order to accomplish visual tasks. In the biological world, organisms need vision in order to recognize their friends and enemies, avoid danger, find food and in general survive. In the world of robots, vision is needed to make them capable of performing various tasks while interacting with their environment. However, recovering the scene and its attributes is not a necessary condition for the accomplishment of visual tasks. Many such tasks can be achieved visually without reconstruction but through the recognition of patterns, objects or situations.

What to recognize is concerned with the questions we pose. The purposive paradigm calls for formulating questions that are directly related to visual tasks, i.e. that have a purpose. Knowledge of 3-D motion is much more than we need to answer the purposive question: Is this moving object coming closer to the observer? Purposive thinking leads us to pose questions whose answers will only help to solve the particular task at hand, and will not be of general use. This level of the paradigm is parallel to Marr's computational theory and makes sure (or rather, tries to insure) that the resulting algorithms will be of minimal complexity.

How to recognize (patterns, objects or situations) is related to the algorithmic level of Marr's paradigm. Qualitative vision calls for the development of algorithms that are simple, robust and based on qualitative techniques, such as comparisons of quantities or discrete classifications. Qualitative vision, which in the past has been wrongly called inexact, makes sense here because it is coupled with purposive vision,

which formulates questions for which qualitative solutions are possible.

To demonstrate the usefulness of the approach we consider visual motion (or navigation) problems, and assume that the observer is active. We describe the preliminary design of Medusa, a purposive and qualitative visual motion machine that can robustly solve many navigational problems without reconstructing the scene.

1 Introduction

The visual sense is potentially as important to machines as it is to humans. Much progress has been made in the past 35 years in developing visual capabilities for computers, but machines still fall far short of humans and animals in their visual performance. This may seem surprising to a lay person or to one who hasn't thought about vision, since vision is so easy for us. On the other hand, it is not surprising to specialists in the fields of computer vision or image understanding, who know how hard the technical problems are. We understand, for example, that solving the problem of structure from motion (or relative orientation, or passive navigation, or kinetic depth) using two frames (two views) is very hard in the presence of noise [Spetsakis and Aloimonos, 1988]. We also understand, to take another example, that the problem of finding depth discontinuities is also very hard, as its solution will probably come from some complex minimization procedure [Shulman and Aloimonos, 1988].

The field of computer vision has become quite a sophisticated discipline, and it is becoming a well defined science. Most of the analyses presented in recent publications begin with a mathematically precise description of the representations operated on and produced by the visual process under consideration. The technical content of image understanding is becoming more sophisticated. Observations of and assumptions about the world are beginning to be expressed in sophisticated mathematical ways. Relationships between smoothly varying quantities give rise to differential equations. Boundary finding is attempted through Monte Carlo techniques [Marr, 1982] which find the most probable value of the real world variable under consideration, or through advanced machinery of functional analysis [Aloimonos and Shulman, 1989a]. After making the image formation process explicit [Horn, 1986]—the first step in making computer vision a scientific discipline—researchers became concerned with geometry (gradient space, perspective invariance). Combining geometry with properties of physical surfaces leads

The support of the Defense Advanced Research Projects Agency (ARPA Order. No. 6989) and the U.S. Army Engineer Topographic Laboratories under Contract DACA76-89-C-0019 is gratefully acknowledged.

to multivariate real analysis, differential and algebraic geometry. Fields such as functional analysis, stochastic approximation, non-standard analysis and more advanced topics are actively used today in research on image understanding. In addition, mathematics that were very recently discovered—such as minimum description length estimation [LeClerc and Fua, 1987] and the regularization paradigm [Poggio et al., 1985]—are actively used in computer vision today, and books examining methodological issues have been published [Aloimonos and Shulman, 1989b].

But the lack of practical systems capable of performing non-trivial visual tasks has led to some confusion in our scientific community. This is partly expressed by the number of panels at recent meetings, workshops and conferences, asking questions such as: Are we doing the right things? How important is this or that? Different schools of thought have groups of researchers who have similar ideas about how one should proceed in order to understand vision. We discussed this in a previous publication [Aloimonos and Shulman, 1989b]. Recent research obliges us to maintain that, while it is beneficial to continue developing visual mathematics because this development will uncover various secrets of perception, we already know enough about the visual process that we can start developing robust machines possessing visual capabilities. But in order to accomplish this, we must change the way we think about vision problems; we have to take a fresh point of view.

In Section 2 we give a short description and criticism of the current state of affairs. In Section 3 we introduce our new paradigm.

2 Status quo: Recovery

Perhaps the most important discovery (or observation) in our field was the reduction of vision to a recovery problem [Horn, 1986]. Indeed, neither objects nor properties of objects (such as shape, color, lightness, etc.) exist inside our brains as such. We only find brain cells there, but these cells must be performing a symbolic descriptive function. In other words, when we see, symbols are somehow computed inside our heads, and these symbols represent things in the outside world.

Each of our eyes works like a camera. Both eyes and cameras have lenses, and where the camera has light-sensitive film, the eye has a light-sensitive retina. The lens focuses an image of the world onto the retina, which then sends messages about brightness values in the image along the optic nerve fibers to the brain. Sets of specialized neurons in the brain can then represent the imaged scene. For example, the level of activity of one of these neurons might correspond to the brightness of the corresponding point in the retina, and hence of the associated point in the scene.

Once a representation of the imaged scene is transmitted to the brain, algorithms running on the neural hardware can extract useful information from the representation. This information is very important for our everyday activities and may be essential for survival. Using visual information we recognize our mates, friends and enemies and we find food. We recognize objects, we

drive vehicles, we walk or run avoiding obstacles, and we pick up things with our hands.

But what is the information in images that is relevant and necessary for accomplishing such tasks? And how is this information extracted? Answers to such questions will constitute the basis for our study of vision.

If more than half of the neurons in our brain are devoted to solving vision problems, it is not surprising that understanding our visual system is not an easy problem. Equivalently, designing a machine that can "see" is a very difficult task.

There exist two general goals for a successful vision system: navigation in a complex environment using vision, and recognition of classes of objects (such as people or trees) in a complex scene. A large proportion of the research on computer vision addresses, explicitly or implicitly, one of these two goals. But achieving these goals presents great difficulties.

These difficulties were realized during the 1960s and '70s after the failure of earlier attempts to build complete vision systems, i.e. systems that used knowledge at all levels including domain-specific information. "In order to complete the construction of such systems it is almost inevitable that corners be cut and overly simplified assumptions be made" [Brady, 1982]. Doing this results in a system capable of performing a limited set of tasks, but not enhancing our general understanding of vision.

At about that time it was proposed [Marr, 1982] that many visual tasks depended on solving the following problem: From one or more images of a scene, derive an accurate three-dimensional geometric description of the scene and quantitatively recover the properties of the objects in the scene that are relevant to the given task. If we can recover an accurate description of our environment, we can navigate and avoid obstacles, and if we can accurately recover the properties of an object (shape, reflectance, color, etc.) we can use them to recognize it.

How can recovery be accomplished in a complex visual environment? By following the general principles for the design of complex systems [Feldman, 1985]. We divide the system into functional components which break the overall task into autonomous parts, and analyze these components individually. We then choose the representations of information used by the components and the language of communication among them. The components are then tested individually, in pairs, and all together.

In a visual system, the components are subsystems that recover specific properties of the scene from images. We call these subsystems *modules*. The majority of computer vision research has been devoted to the study of such modules [Horn, 1986] and their integration [Aloimonos and Shulman, 1989b]. The study of human and animal perception provides evidence as to the nature of the modules. For example, one source of evidence for the existence of modules in the human visual system is the study of patients with disabilities that come from brain lesions. On the other hand, psychophysicists perform experiments in which a particular module of the human visual system is "isolated", such as Julesz's [1982] experiment on stereoscopic fusion without

monocular cues, Land's [Land and McCann, 1971] demonstration of the computation of lightness, Gibson's [1950] experiments on the perception of shape from texture, etc. These studies suggest that cues such as shading (image intensity variation), texture (distribution of surface markings), contours and outlines (image discontinuities), color, motion and stereo are very helpful in recovering properties of the scene from images. In computational vision, names have been given to many of these modules: Shape from shading, shape from texture, shape from contour, shape and depth from stereo, structure from motion, direction of light source from intensity, physical discontinuities from intensity discontinuities, motion from image intensity derivatives, etc. Analyzing vision is then reduced to analyzing how the individual modules work and how they interact.

In [Marr, 1982] a methodology was developed for analyzing visual modules. According to the *Marr paradigm*, in order to understand a visual process we must consider three levels:

- (a) The level of *computational theory*. We should develop, through rigorous mathematical treatment, the relationship between the quantity to be computed and the observations (image(s)). After this computational theory is developed, we will understand whether or not the problem has a unique solution.
- (b) The level of *algorithms and data structures*. After the computational theory has been developed, we should design appropriate algorithms and data structures that, when applied to the input (image(s)), will output the desired quantity. If the problem has a solution, there are probably many ways to find it. This level is concerned with choosing ways that are efficient, robust, etc.¹
- (c) The level of *implementation*. After the two previous levels have been developed, we must implement the algorithm on a machine (serial or parallel) in order to obtain a working system.

Subsequent important research along the lines of the Marr paradigm continued to consider vision as a recovery problem. During the past 15 years a plethora of elegant mathematical theories describing various modules has appeared. Unfortunately, there is a disconcerting lack of visual systems which perform well in real-world environments, particularly when compared to the amount of mathematical theory published on the subject. There seem to be several reasons for this.

One reason is that extracting useful visual information from images probably involves a very large amount of computation. The visual cortexes of animals that perform complex visually moderated behaviors contain millions of neurons, each of which performs computations which require thousands of computer steps per second to simulate, and possibly many more. Much of this capacity is

probably necessary to carry out whatever cortical image processing occurs.

A somewhat related reason is the perception that practical results will eventually flow from a successful theory rather than vice versa.² This probably has more to do with the lack of any practical systems to work with than with philosophical conviction, since historically, empirical engineering applications or unexplained observations have preceded theoretical developments at least as frequently as the reverse. If there were suddenly to appear a number of machine vision systems working robustly in different real-world domains, it is quite probable that theories explaining their commonality would soon appear.

There is a third reason that may explain the dearth of examples of working vision systems, which is that the generally accepted goals for such systems may be misplaced, or at least over-ambitious. The two commonly held touchstones for practical vision systems, recognition and navigation, are high-level objectives. If both were achieved, automatic systems would have many of the capabilities of the human visual system.

Given the lack of success in developing systems which realize either of these goals in a robust manner, it would appear reasonable to consider simpler problems; many researchers have followed this avenue by working on a very specific problem such as an industrial application. However, these approaches do not enhance our understanding of vision in general.

Another problematic aspect of the recovery (or reconstruction) school of thought is the fact that visual computations are reduced to finding the value of some real quantity and usually the success of the visual task relies on the accuracy of the first or second decimal digit of that quantity. As a result, most machine visual tasks are unstable. A slight error in the input is enough to destroy some computations.³ How can we then achieve robust visual computations that can be reliably used for accomplishing various tasks?

There is, luckily, another way to view visual computations. We can consider simpler problems! Although if we can recover the world we will be able to achieve many tasks, it is not always necessary to perform this recovery. Rather, we should ask ourselves the question: What is vision for? [Ballard, 1989] We will then immediately realize that we need vision in order to accomplish tasks that are essential for our survival, as already described. But to carry out the task, it is not clear that we need to recover the world and its properties. To give examples from biological vision, a housefly can maneuver visually in three dimensions in a complex environment without striking obstacles; a bee can recognize and return to a particular location in its environment from an arbitrary (nearby) starting point, an ability referred to as

²This point of view was suggested by Nelson [1988].

³We analyze the case of structure from motion later.

homing [Nelson, 1988].

When we study visual abilities, we should study them in a purposive manner. We need to keep in mind the question: What am I going to use this visual ability for? What are the tasks that can be performed using it? Thinking in a purposive manner will convince us that when we want to go from place A to place B in a room with many people moving around, we just need to avoid obstacles; it is not necessary to reconstruct our extrapersonal space and thus know that the person in the corner is smiling! Clearly, if we could reconstruct the room and the moving people, our task of going from A to B would be very simple. But it is obvious that reconstruction is not necessary.

If we study vision in a purposive, or utilitarian manner [Ramachandran, 1989], or an animate manner [Ballard, 1989], the problems that we formulate are much simpler, since they are relevant to the task at hand, and since they are simpler they can be solved by qualitative techniques that exhibit robustness properties, as we shall see in the next section.⁴

3 Purposive and qualitative vision: A new paradigm

Although the foundations of purposive and qualitative vision lie in mathematical and engineering considerations,

⁴Some historical comments are in order here. The recovery school of thought was created by Marr and his colleagues at MIT (Poggio, Horn, Binford, Ullman, Brady), and it was the first serious attempt to transform image understanding into a scientific field. Some researchers have always questioned this approach. Some people have devoted their research to building specific systems, but they were so preoccupied with this that they forgot to worry about general principles. But during all this time, many workers in the field (Feldman, for example) have felt that vision should be goal based, goal driven. However, the goal driven concept has been misused as a vehicle for expensive experimentation, whose only fruits have been the realization that vision is a very hard problem. Since the middle of the 1980s, and sporadically before then, researchers have been interested in qualitative vision. They felt that techniques and algorithms whose success depended on the decimal digits of a quantity had to be replaced by new methods, qualitative in nature, that had the potential for robustness. However, qualitative techniques were tried for some time without much success, primarily because the researchers who applied them were thinking in a reconstructionist paradigm. In 1989, Ballard wrote a paper on "Animate vision" [Ballard, 1989], where he claimed that vision should be active (as we had suggested in 1987 [Aloimonos et al., 1988]), and that it should have a goal. The view of V.S. Ramachandran [1989] on a utilitarian theory of perception, and the views of D. Regan [Regan], suggest that the human visual system may consist of a large number of simple processes, each devoted to solving a particular task. (This is also supported by recent work on visual illusions [Aloimonos and Huang, 1990].) Consequently, it makes sense to study computer vision in a purposive and qualitative manner. Purposive, in order to formulate the right (simple) problems; and qualitative, in order to obtain robust solutions. In any case, the problems are formulated in such a manner that it makes sense to solve them qualitatively. All this gave rise to the theory of purposive and qualitative vision described in this paper. Research of this flavor has also appeared lately in the work of the Genoa vision group [IEEE, 1989] and the work of Nelson [Nelson, 1988], Weinshall [Weinshall], Poggio and Edelman [Poggio and Edelman, 1989], and Zucker [1989].

it seems to be consistent with evolution. It appears that some organisms developed light sensitive skins at some places on their bodies and learned to respond to light and dark. Other visual capabilities evolved because of specific needs. Accepting that the ultimate goal of an organism is survival, visual abilities should have evolved in such a way that they served survival purposes. Thus, visual abilities for avoiding danger, recognizing food, recognizing mates, friends and enemies developed. But although some of these abilities were based on common principles (for example the ability to intercept a moving object and the ability to avoid a moving object are both based on the structure from motion module), they were possibly developed at different times and it is probable that they are implemented by separate hardware. From this point of view we may expect that the machinery of the brain devoted to vision consists of various independent processes (which of course communicate) that are devoted to the solution of specific visual tasks. This also seems to be the view of leading neuroscientists [Regan] and connectionists [Ballard, 1989].

Let us now restrict our attention to problems related to visual motion analysis, in order to make things simpler to understand. Consider a robust vision system (of the future) that "understands" (can handle) visual motion. According to the "state of the art" in the recovery school, such a system is currently envisioned to work in two stages, as follows:

- (a) First, from successive dynamic images, retinal motion (or correspondence or optic flow) is computed. Then a segmentation of the various independently moving objects in view is performed, based on the estimated optic flow.
- (b) Second, algorithms are applied to the various parts of the image-motion field, in order to estimate the 3-D motion and the shape of the various parts of the scene. Such algorithms comprise the so-called "structure from motion" module.

It is clear then that if the vision system under consideration can robustly and in real time perform the above two stages, then it will be able to perform many visual tasks, such as passive navigation, obstacle avoidance, tracking, prey catching, and many others. But, although the above two problems have attracted a lot of attention in the past ten years and we have seen a large number of impressive and mathematically elegant results, both problems are far from being solved.

The first problem amounts to recovering a two-dimensional function (the flow field over the image) from a small number of constraints. Thus the problem is ill-posed, as is the case with most visual reconstruction problems. The difficulty of the problem is amplified by the fact that the function we need to recover is not "nice", i.e. it contains corners—its derivatives are not continuous.

Only in the past few years have we started the development of mathematics for the recovery of functions containing discontinuities in the case of underdetermined problems. We described them in a previous paper [Aloimonos and Shulman, 1989b]. However, both

probabilistic [Geman and Geman, 1984; Marroquin, 1985; Mumford and Shah, 1985; Blake and Zisserman, 1987] and deterministic [Shulman and Aloimonos, 1989b] approaches are handicapped by the presence of various difficulties.

For example, in the probabilistic formulation we face the difficulty of computing priors. Other approaches that require the minimization of a complex functional take a long time to converge, and it is very difficult to estimate the various parameters involved in the model [Aloimonos and Shulman, 1989b].⁵ It has also become apparent that such reconstruction techniques, if they are to be of use in general situations, must be coupled with learning procedures, and although some research along these lines has appeared [Aloimonos and Shulman, 1989a], the problem is far from solved.

The second problem has to do with the computation of structure and 3-D motion using the results of the first step. Although this problem has seen a lot of theoretical development, it is far from being solved in the presence of noise [Aloimonos, 1990].⁶

We are facing two alternatives.

The first alternative is to continue our research on recovery, to improve our mathematics, to try to understand why existing approaches are unstable and bring in our knowledge of statistics and engineering, in order to develop provably optimal estimators of structure from motion and in order to introduce noise remedy concepts, such as redundancy for example. Of course, the hope here is that our work will converge to the best possible structure from motion module and that this module will be good enough (i.e. robust), or we might be surprised to find that the best is not good enough for some tasks.

The second alternative is to reconsider our viewpoint about the recovery paradigm, and work "around" the problem. In simple words, this alternative suggests *not solving* the abstract structure from motion problem, not developing the structure from motion module. Instead, it suggests that we must ask the question: What tasks will I perform, if I have a structure from motion module? After the tasks have been identified, this alternative suggests that we should solve them directly and not as an application of a general principle. We should solve directly, for example, the problem of avoiding obstacles. We should ask: Is this moving object coming closer to me or not? If it is coming closer, where is the focus of expansion (FOE)?⁷ Is it inside the boundaries of the image or

outside? If the FOE is in a given area, does this mean that the moving object will hit me? If it is going to hit me, how long will it take with respect to my reaction time? Can we solve such a collection of problems in a robust manner?

If we can solve such problems directly, the structure from motion module is no longer needed.⁸ Moreover, because we now ask simple questions that have a small numbers of possible answers, the potential exists that we will be able to achieve robust solutions, since our approach is qualitative.

We thus see a new paradigm emerging: that of purposive and qualitative vision (which should of course be active). In this framework, one does not look at a vision system as a collection of modules whose purpose is to reconstruct the world and its properties and thus provide information for accomplishing various tasks. Instead, one looks at a vision system as a collection of processes, each (or a group) of which solves a particular visual task. If we look at the computer vision field in this way, in effect we are considering vision, not in isolation, as the recovery school of thought does, but as a part of a bigger process in which vision is used as a front end.

How can we, in this paradigm, generate new research problems?

The answer to this question comes directly from purposive (or utilitarian) thinking. In this new framework we think of vision not as an end in itself, but as part of a bigger process that performs tasks. If the tasks are complicated, we decompose them into simpler tasks, and solve the simpler ones. In other words, it is the task itself that suggests what problems need to be solved. If, for example, we need to construct a machine that can guard an area, the machine must be able to identify anything moving, track it, and intercept it (and possibly also recognize it). These are three different tasks, according to the paradigm of purposive vision. But according to the reconstruction paradigm, these tasks are applications of the structure from motion module.

We cannot study the general problem of vision, not tied to any applications. The reason this is not possible is that we regard vision as a part of a bigger process. If we wish to study vision in general, we should study the tasks that organisms possessing vision can accomplish. This is what we do in this paper. We study vision problems in the paradigm of purposive (and qualitative) vision.

In the rest of this paper we describe how one can study navigational problems in the framework of purposive vision. We do this through the description of a

⁵In a recent AFOSR Workshop on the "Encounter of Mathematics and Computer Vision", organized by Profs. R. Bajcsy and P. Lax, University of Pennsylvania, May 21-23, 1990, many more difficulties were pointed out.

⁶That also seemed to be also the consensus in the special panel on "Visual motion: Current and future problems" that took place during the 10th ICPR, Atlantic City, NJ, June, 1990.

⁷The FOE is basically the point where the moving object will hit the image plane of the observer.

⁸We do not mean that this module and its supporting theoretical research become obsolete. On the contrary, such research, which has become highly sophisticated nowadays (see Section 4), will contribute an immense amount to photogrammetry, cartography and other visual reconstruction problems, such as teleconferencing, for example. What we mean is that if we can solve all the above-mentioned problems, then the structure from motion module won't be needed for an autonomous "seeing" machine that can perform a variety of navigational tasks.

machine that we are building using results from our recent research. This machine, through simple processing of images, very robustly—because of the qualitative form of the solutions—performs a nontrivial number of difficult navigational tasks, without knowing anything about “structure from motion”, and without reconstructing the world; everything happens through simple processing of a large amount of data.

4 Medusa: A purposive and qualitative active visual motion machine

While we continue our visual motion research in the reconstruction school of thought by concentrating on robustness issues, we have started developing a set of processes that are capable of accomplishing various visual tasks. At the same time, we are designing Medusa, a simple active vision machine, that contains all the qualitative visual processes which we are developing. Medusa is so simple that every new ability we are developing can easily be added to her.

Medusa has an active camera system, inertial sensors, and a hand which is visible from the eye (camera), and she can move around in the environment. As Medusa moves (she collects images at video rate), she is capable of accumulating many dynamic frames (views). For the purposes of this paper, we assume that Medusa has available a set of successive image frames. Figure 1 contains a schematic description of Medusa's “brain”. Some of her parts will be explained subsequently.

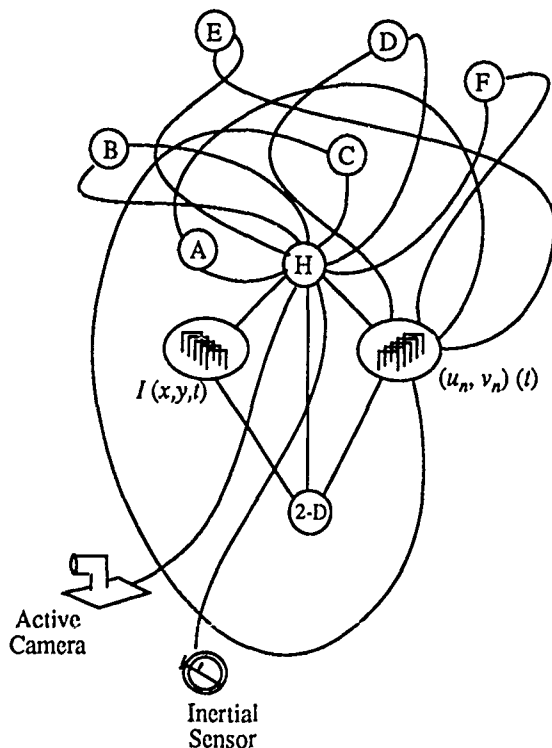


Figure 1. A schematic description of Medusa

The input to Medusa is the series of images $I(x, y, t)$, on the left. The central data structure, shown on the right, consists of an evolving *normal flow field*—i.e., the

spatiotemporal derivatives of the image intensity function. The small circles represent the following visual processes:

- ② From the series of images, computes a series of normal flow fields.
- Ⓐ Answers the following question: Is anything moving independently of me? If so, where is it on the image?
- Ⓑ Answers the following question: Is this moving object getting closer to me? Or, which parts of the image correspond to parts of the scene which I am getting closer to?
- Ⓒ Performs the task of keeping a moving object in the center of the visual field (tracking) by appropriately rotating Medusa's eye.
- Ⓓ If the moving object is getting closer, finds out if the focus of expansion lies inside the image. If so, determines an area in which the FOE lies. Using Ⓒ and some crude information about depth, finds if the moving object will hit Medusa.
- Ⓔ Tells Medusa's motors how to move in order to intercept a moving object (prey catching), using all the previous processes.
- Ⓕ Performs kinetic stabilization for Medusa (passive navigation).

: : : :
: : : :
: : : :

Finally, Ⓕ is what we call the homunculus of Medusa, i.e. its central controller. Ⓕ is the master control unit that communicates with all the other processes, knows the tasks that Medusa intends to execute, knows which processes, in what order, are needed for every task, and synchronizes processes Ⓐ – Ⓔ. Clearly, Ⓕ is a primitive control, as is Medusa herself. While it is true that the control of the processes can be distributed, this is of no concern to us yet.⁹ Also, Medusa has a set of motor units that control the motors of her parts and communicate with the central control, but they are of no concern to us here. She also has inertial sensors for measuring her angular acceleration, from which her rotational motion can be driven. Medusa can be easily augmented and made more powerful, especially if she acquires a second eye.

Medusa demonstrates that one may construct powerful machines that solve a series of simple problems, and establishes an alternative way of thinking about image understanding in general. Medusa cannot compute exact

⁹To fully understand the control problem, we need to know the mathematics of cooperating agents [Minsky, 1988]. But to accomplish some nontrivial tasks we don't need complicated control; simple methods are enough.

3-D motion. As a matter of fact, she has no concept of it. But she doesn't need it to carry out any of her tasks.¹⁰

We now proceed with a short explanation of some of the processes. In the course of the analysis the reader will understand some of the tools of qualitative vision. Medusa doesn't use flow or correspondence. She uses image gradients only.¹¹

4.1 The input: Finding normal flow

The normal flow is the only representation of image motion that can be robustly computed. If $f(x, y, t)$ is the image intensity function as it changes through time, and (u, v) is the motion field (the image velocity as a projection of the 3-D motion), then

$$f_x u + f_y v + f_t = 0 \text{ or}$$

$$(f_x, f_y) \cdot (u, v) = -f_t$$

where the subscripts denote partial differentiation and " \cdot " denotes the inner product of vectors.

From the above equation, it follows that we can compute the projection of the optic flow (u, v) on the image gradient direction (f_x, f_y) at every image point. The details of the implementation are not given here. The series of images is first appropriately smoothed before the derivatives are computed. For the purposes of this paper we assume that Medusa has available a series of normal flow fields, $[u^n(x, y, t), v^n(x, y, t)]$. It may happen that there is not enough information at a given image point to compute the gradients, because the values are very small. We do not compute the normal flow at such a point.

4.2 (A) Detecting independent motion

Medusa has the capacity to move around in her environment. But regardless of whether she is static or she moves, she needs to be able to detect anything moving independently around her. This problem has been studied in the literature and various solutions have been proposed (see [Nelson] for a survey).¹² There is, however, a very strong geometric constraint, first exploited by Nelson [Nelson], that makes the problem easy.¹³ We consider it for the case where Medusa is moving (if she is static, the

¹⁰Our approach is not to be confused with Brooks' paradigm of achieving AI through building robots. Brooks' program is very expensive [Brooks, 1986]. In addition, he studies specific problems, while we study problems of environmental invariance; we study visual abilities.

¹¹Because our algorithms are based on image derivatives they have some similarities, but only in spirit, with direct motion algorithms [Aloimonos and Brown, 1984; Negahdaripour, 1986; Horn and Weldon, 1989].

¹²The solutions are generally quite complicated; they depend on clustering of flow fields and they don't work for multiple moving objects. The most robust algorithm for detecting an independently moving object in a series of images taken by a moving observer is probably that described in [Burt et al., 1989].

¹³Actually it was first mentioned by Marr [Marr, 1982].

problem is much easier; she just detects normal flow and that is enough to detect the presence of something moving, as well as its position on the image). If Medusa moves, then the flow field that she gets is due to rigid motion if the scene is static. If, however, there is something in the scene moving independently, then the resulting flow field will be due to nonrigid motion. The question then becomes: can we find out if the changing image is due to nonrigid motion, using just the normal flow?

Medusa has an approximate idea of how she is moving. Her inertial sensors provide her with bounds on her rotation (A, B, C) . Let us assume that $A \in [A_{\min}, A_{\max}]$, $B \in [B_{\min}, B_{\max}]$ and $C \in [C_{\min}, C_{\max}]$. There are also bounds on her translation (U, V, W) , $U \in [U_{\min}, U_{\max}]$, etc; these bounds come from the motor units and could also come from process (D).

Consider now an image point (x, y) , where the normal flow (u^n, v^n) has been measured. Also consider the flow space (u, v) in Figure 2. The actual flow at (x, y) is $(u, v) = (u_R, v_R) + (u_T, v_T)$. The rotational parts u_R, v_R are polynomials in x, y and linear in A, B, C [Horn, 1986]. On the other hand, the translational flow (u_T, v_T) is

$$u_T = \frac{-u + xW}{Z}, v_T = \frac{-V + yW}{Z}$$

where Z is the depth of the point whose image is (x, y) . Assume further that Medusa has a constraint on depth ($\alpha < Z(x, y) < \beta, \forall (x, y) \in \text{image}$).¹⁴ If we consider U, V, W as fixed, then the translational flow will lie somewhere in an interval CD (Figure 2), and consequently the actual flow will lie somewhere on a line segment BE . So if Medusa knows her motion exactly, then she knows that the actual flow at a point must lie on a line segment in flow space, due to the depth constraint.

Now consider Figure 3. If the actual flow lies on a segment AB , where could its corresponding normal flow be? Clearly, if the actual flow is at point A , the normal flow there could be anywhere on the circumference of the

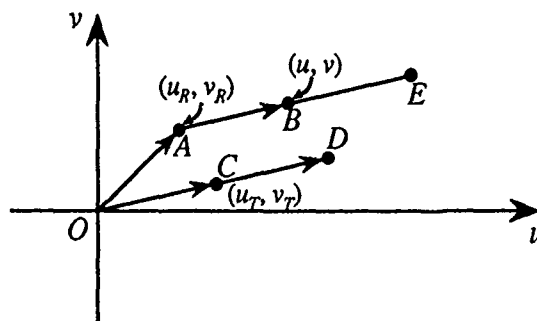


Figure 2.

¹⁴This comes from another process which we don't describe here.

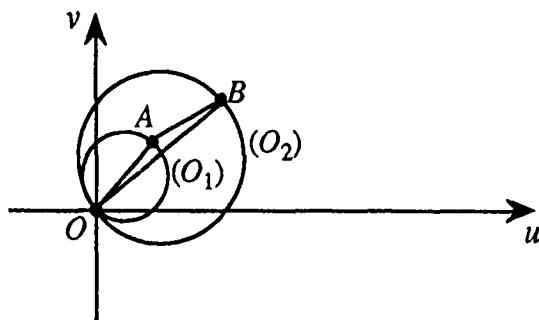


Figure 3.

circle with diameter OA . As point A moves, circle (O_1) moves towards circle (O_2) . The union of all those circles (the circles with diameter Ox where $x \in AB$) defines an area Ω (the forbidden area). If the normal flow at point (x, y) does not lie inside Ω , then Medusa knows that along the ray connecting the nodal point of her eye with the image point (x, y) , there is something moving independently.

In actuality, the area Ω is slightly more complicated, because Medusa doesn't know her motion exactly. This means that the rotational flow (point A) lies inside a rectangle. Similarly, the translational flow part (point C) lies inside another area. As a consequence, the actual flow lies inside a parallelogram whose points are the sums of vectors \vec{OA} and \vec{OX} . The area Ω is again defined analogously.

Thus every point (x, y) in the image has an area $\Omega_{x,y}$ associated with it. If the normal flow at a point does not fall inside Ω , Medusa knows that it is due to an independently moving object. If it falls inside, Medusa doesn't know. But luckily, Medusa is an active observer, and the forbidden area depends on her own motion. If her motion changes, so does the forbidden area at every point. Thus, if the normal flow at (x, y) falls inside $\Omega_{x,y}$ at time t (see Figure 4)—and this means that no inference about independent movement is possible—at the next time instant, Medusa can totally change the forbidden area by making a controlled saccade. If the normal flow (point A in Figure 4) still remains inside, Medusa infers that no independent movement is taking place. But if A falls outside after a saccade, then independent movement is signaled.

The details of the implementation and a control strategy that guarantees correctness are still research issues. It is important to point out that this technique is qualitative (checking if a point lies inside or outside a region) and it can be achieved through simple inequality tests (checking if one quantity is larger than another one).

Figure 5 shows some experimental results. Figure 5a shows a scene containing moving cars on a (model) road. The white dots in Figure 5b are points where Medusa detected independent motion. The scattered points that do not lie on the cars were detected because the computation

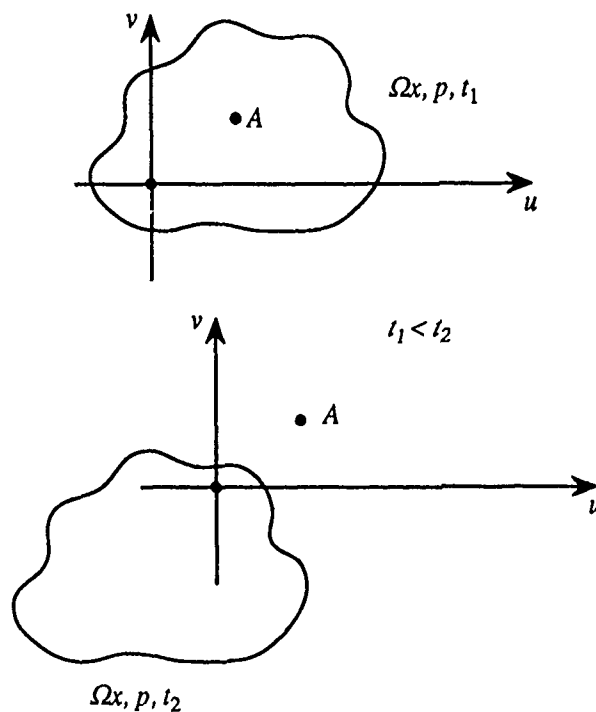


Figure 4.

of the image derivatives was not optimized.

4.3 (B) Is the object getting closer?

A robust method of deciding whether an object is getting closer was described in [Nelson and Aloimonos, 1989]. This method is based on the following proposition: If the distance between the sensor and an object projecting to point p on the image is decreasing, that is, the perpendicular component of the relative translation is positive at p , then there exists a direction ϕ for which the directional divergence $D_\phi f(p)$ is positive, where f is the flow at p and D_ϕ is the directional divergence operator.

In [Nelson and Aloimonos, 1989] a system was constructed that, based on the divergence of the flow field along the image gradients (f_x, f_y) , was able to detect objects whose distances from the sensor were decreasing. The technique works well because divergence is a persistent feature, and its sign is enough to determine whether a given part of the scene is getting closer.

Using this technique, Medusa can always detect if she is on a collision course with an obstacle, but sometimes she thinks she is on a collision course when she isn't.

Her dilemma can be resolved by using process (D), but in any case, at worst Medusa might do some unnecessary maneuvering to avoid something that she wrongly thinks is coming closer, but she will never collide with an obstacle.

Figure 6a shows a test of this technique in which the camera moved toward an obstacle (a piece of tree bark) seen against a distant textured background (a Paisley print). Figure 6b shows the corresponding hazard map

(white denotes points of high flow field divergence). Evidently, such a hazard map should be quite adequate for Medusa's purposes.¹⁵



Figure 5a.

4.4 © Keeping the object in view

Tracking an object (smooth pursuit) has been studied from a reconstructionist standpoint and various algorithms have been developed that worked reasonably well for scenes that fitted their assumptions [Aloimonos and Tsakiris, 1988]. It is, however, hard to implement a general control regime that smoothly rotates the camera in order to track an object. But, since Medusa is an active observer, she can keep an object visible through a sequence of saccades. Her control regime rotates the camera so as to minimize the vector defined by the origin of the image plane and the center of mass of the points in the image that process (A) detected as moving independently. Thus, although Medusa cannot yet implement smooth pursuit, she can use saccades to keep a moving object always in view.



Figure 5b.

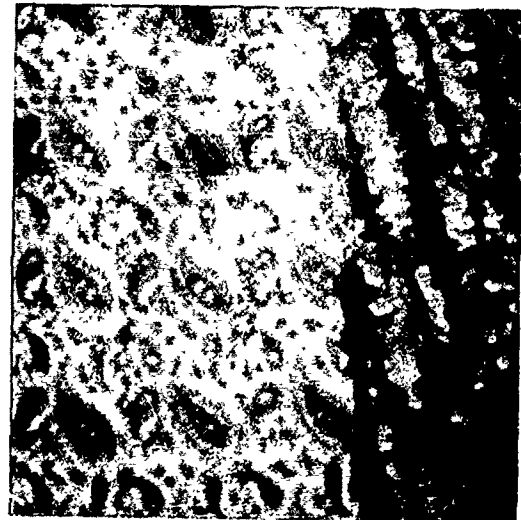


Figure 6a.

4.5 (F) Finding the focus of expansion

If Medusa is only translating, it is easy to find the FOE (focus of expansion) if it lies inside the image; and if it lies outside, it is easy to find in which direction it is (we can bound it).

Indeed, consider Figure 7, where we examine the normal flow at a point. Since there is only translation, it is certain that the actual flow vector at point (x, y) will lie on the side of line e which contains (u_n, v_n) , and it is also certain that the FOE lies on the other side of e . We can thus vote for every point on the side of e opposite (u_n, v_n) as a candidate FOE. We repeat this for all points at which we have measurements of the normal flow; this can be done in parallel. The area with the highest number of votes (the intersection of all the half-planes) contains the FOE. In our experiments with real scenes, this area was generally very small (a few pixels). In any case, because

¹⁵If we can derotate and get constraints on the translational flow, then we can show, under certain assumptions about the angle of the lens, that if P is a point on the image and the sensor is translating along the line OP with velocity v_{perp} towards a surface at distance Z , then

$D_{\phi}(f_{\text{perp}}) = \frac{v_{\text{perp}}}{Z}$ independent of ϕ and the orientation of the surface, where D_{ϕ} is the directional flow divergence operator, and f_{perp} is the part of the translational flow due to the velocity v_{perp} at P . Since $\frac{Z}{v_{\text{perp}}}$

is related to "time to collision", this gives us another qualitative technique for finding if time to collision is smaller or larger than some threshold. Medusa has this ability. Although the translational flow is also due to the component of translation parallel to the image, we can make use of the above relation [Nelson and Aloimonos, 1989].



Figure 6b.

the area is convex, even if it lies partly outside the field of view we know that the FOE must lie in an angular sector bounded by the tangents to the area.

If Medusa is both translating and rotating, since she is equipped with inertial sensors, she can detect her rotation, or at least find bounds on it. This means that the component of the flow due to rotation is constrained. Since the rotational flow is constrained, the translational flow becomes constrained too, and, we can again locate the FOE.

We now describe a qualitative technique that solves the problem visually. Consider any point (x, y) on the image plane, and let $\vec{f}_1, \vec{f}_2, \dots, \vec{f}_n$ be the normal flow vectors at (x, y) at times t_1, t_2, \dots, t_n . The underlying assumption is that the motion of the object remains constant during the time interval $[t_1, t_n]$. Medusa would like to check whether (x, y) is the FOE. Consider the actual flow (u, v) at (x, y) . Then

$$u = \frac{1}{2}(U - xW) + u_R$$

$$v = \frac{1}{2}(V - yW) + v_R$$

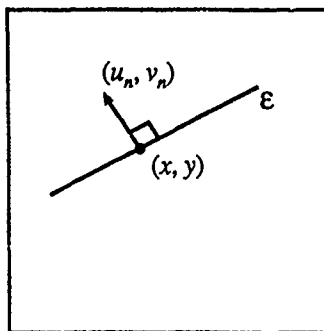


Figure 7.

As time changes, u_R and v_R remain the same. The quantities $(U - xW)$ and $(V - yW)$ remain the same too. Only Z changes in an unpredictable way (most points come closer but not necessarily). If at a particular point (x, y) , the flow does not change its x -coordinate, then $x = \frac{U}{W}$. Similarly, if the flow's y -coordinate does not change, $y = \frac{V}{W}$. Thus if the flow at (x, y) remains constant throughout the period $[t_1, t_n]$, excluding degenerate cases such as a planar surface translating parallel to the image plane, an object only rotating, and the like, then (x, y) is the FOE.

Figure 8 shows that such a point can be found using only normal flow measurements. If the optic flow at (x, y) remains constant, lines $\epsilon_1, \epsilon_2, \epsilon_3, \dots$ should pass through the same point, and this can easily be checked with a robust technique. In a similar way, one can bound the FOE if it falls outside the image. Once the FOE has been bounded, the translational flow becomes constrained, which in turn constrains the rotational flow and thus the rotation.

It must be emphasized that in the calculations used above in all three techniques, the measurements are done only in the areas that do not belong to independently moving objects (ability A). Figures 9a and 10a show two complex laboratory scenes to which these techniques were applied. In Figure 9 the motion was translation along the optical axis; Figure 9b shows the top part of the scene in two successive frames (note that the edges have shifted). Figure 9c shows the FOE area computed using only 50 halfplane constraints, and Figure 9d shows it using all the constraints in the image; this locates the FOE very accurately. In Figure 10 the motion was along a curve in the horizontal (x, z) plane, starting out to the left $(-x)$ and curving back toward the z axis. Figure 10b shows the top part of the scene in two frames; note that the edges are both shifted and rotated. Figures 10c and 10d show the FOE area using all the constraints in the image as estimated at early and late stages in the image

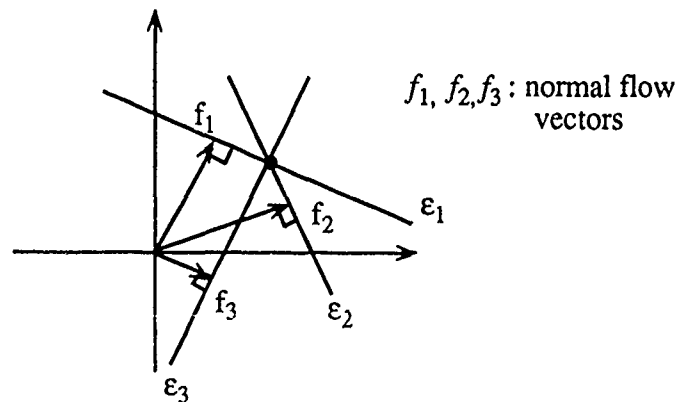


Figure 8.

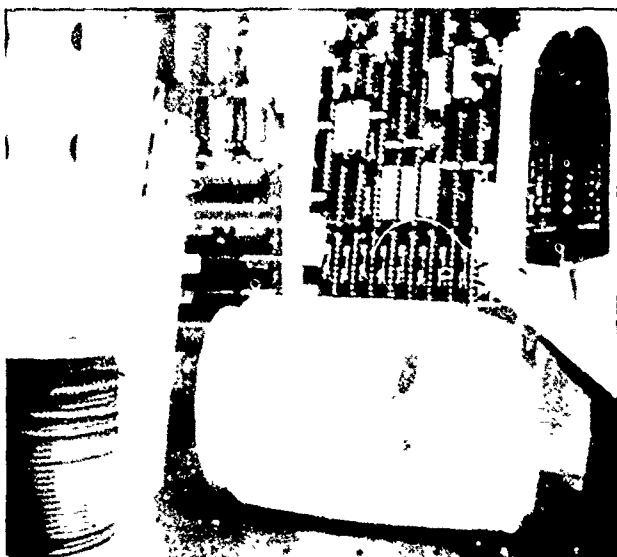


Figure 9a.



Figure 10a.

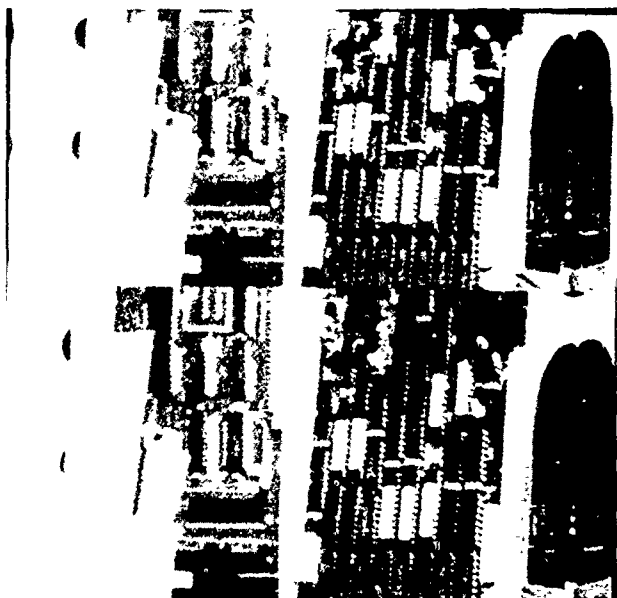


Figure 9b.



Figure 10b.



Figures 9c and 9d.

sequence; both estimates tightly constrain the position of the FOE.

4.6 ④ Is the object going to hit me?

This is an important question, because a correct answer will eliminate false alarms. If process ② decides that the



Figures 10c and 10d.

distance between Medusa and an object is decreasing, that might result in an alarm, but the object and Medusa may be not on a collision course.

To answer question ④, Medusa fixates the moving object (so that the optical axis of her camera passes through it) after she detects it using ability ①. Using techniques similar to those described in Section 4.5,

Medusa computes the direction of the object's translation (ignoring its rotation around itself). Then, using bounds on the object's distance (which can be obtained from other sources), she can determine whether or not she is on a collision course with the object. (Collision is impossible if the object is sufficiently far away, and it is unavoidable if the object is sufficiently close.)

5 The rest of Medusa

It is impossible to completely describe Medusa within the bounds of this paper. We only hope that the reader understands her basic structure. It consists of a set of processes, all operating on the spatiotemporal gradients of a sequence of images. These processes perform basic tasks. She has control units that allow communication of the results of the processes, and also wake them up or suspend them. Medusa's method of qualitative hand/eye coordination is described in the paper by Hervé et al. in these Proceedings.

6 Conclusion

We have presented a paradigm for studying vision problems, namely that of purposive and qualitative vision. Purposive means that we study vision in conjunction with a purpose, a goal, a task. We do not consider vision alone, but rather as a part of a bigger system that *does* something. If we do this, we discover that we can solve many problems without reconstructing the world and its properties. Many of the tasks that machines need to perform on the basis of vision are sequences of subtasks possessing environmental invariance, like the subtasks performed by Medusa (A, B, C, D, E,...). Solving such problems is the subject of purposive (or utilitarian) vision. Qualitative means that we pose the questions in such a manner that the answer is either yes/no, or a discrete classification. This often allows us to exploit the geometric nature of the problem and usually find a simple answer. Medusa's successful operation demonstrates the usefulness of purposive and qualitative thinking.

We emphasize that purposive and qualitative vision is not just a bunch of hacks! Understanding what the individual subtasks should be (Medusa's A, B, C,...), is not easy. The final product, a nontrivial working vision machine, consists of a set of interconnected processes, each of which is dedicated to solving an easy problem; but all of them, when coordinated, can solve very hard problems. Achieving such a system will come, not from ad hoc research, but from a systematic study, following the principles of the Marr paradigm. The difference is that instead of the reconstructionist modules (shape from x , structure from motion, etc.), we have purposive and qualitative recognitionist modules, i.e. abilities that recognize patterns, situations or objects, well enough to perform appropriate actions. We emphasize again that our guiding principle is: What are the tasks that the vision system must carry out? We must define the tasks in such a way that they are simple. Medusa demonstrates that it is possible to achieve sophisticated visual abilities using simple, qualitative techniques.

Medusa needs to acquire knowledge about the world, as well as intentions. When she has both, she will exhibit very intelligent behavior. We may also want to couple vision with other sensory modalities (vestibular, for example) in order to increase robustness.

The paradigm (introduced by Brooks) of achieving artificial intelligence through building robots is diametrically opposite the recovery school of thought; they are at the ends of a spectrum along which all research in the field falls. Purposive and qualitative vision falls in the middle. It draws from both paradigms, while avoiding some of their problems.

Suppose we ask a human to identify an object (or perform a visual task), and we measure the time T between the instant that the object is displayed (or the task started) and the instant at which it is identified (or the task ended). Let the average time for a neuronal firing, i.e. the time that it takes a neuron to perform a computation and pass the result to its neighbors, be t . The quotient T/t is essentially the number of computational steps performed by the brain in order to identify the object. Amazingly, we find that this number is only a few hundred! Existing computer vision systems that perform non-trivial tasks require millions or billions of steps on a serial computer. There exist "massively parallel" computers; for example, the Connection Machine has 64,000 processors. But we don't yet know how to design parallel programs that can solve complex visual problems in a few hundred steps.

What will happen as our technology improves? It is possible that faster machines, along with improvements in algorithms (for example, integration of visual modules), will lead to high performance visual systems. However, it is also possible that humans (and animals) have fast vision systems not only because they process images in parallel, but also because they use different algorithms, which are more simple and qualitative, and (perhaps) do not perform reconstruction of the scene as a preliminary to carrying out all visual tasks.

This makes the paradigm of purposive vision very appealing, in the sense that it can bring faster progress to the field of image understanding. It can allow us to construct smart machines that will meet the demands of the market, something absolutely necessary for transforming computer vision into a well founded, mature engineering discipline.

Acknowledgement

Thanks to Azriel Rosenfeld for his advice and constructive criticism.

References

- [Aloimonos, 1989] J. Aloimonos, "Unification and integration of visual modules", *Proc. Image Understanding Workshop*, 1989.
- [Aloimonos and Brown, 1984] J. Aloimonos and C.M. Brown, "The relationship between optical flow and surface orientation", *Proc. ICPR*, 1984.

- [Aloimonos and Huang, 1990] J. Aloimonos and L. Huang, "Motion-boundary illusions and their regularization", Technical Report CAR-TR-495, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, 1990.
- [Aloimonos and Shulman, 1989] J. Aloimonos and D. Shulman, "Learning early vision computations", *J. Opt. Soc. Am.*, A6(6), 1989a.
- [Aloimonos and Shulman, 1989b] J. Aloimonos and D. Shulman, *Integration of Visual Modules: An Extension of the Marr Paradigm*, Academic Press, Boston, 1989.
- [Aloimonos and Tsakiris, 1988] J. Aloimonos and D. Tsakiris, Technical Report CAR-TR-390, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, 1988.
- [Aloimonos et al., 1988] J. Aloimonos, I. Weiss, and A. Bandopadhyay, "Active vision", *Int'l. J. Comp. Vision* 1, 333-356, 1988.
- [Ballard, 1989] D.H. Ballard, "Animate vision", *Proc. IJCAI*, 1989.
- [Blake and Zisserman, 1987] A. Blake and A. Zisserman, *Visual Reconstruction*, M.I.T. Press, Cambridge, MA, 1987.
- [Brady, 1982] M. Brady, "Computational approaches to image understanding", *ACM Computing Surveys* 14, 1982.
- [Brooks, 1986] R.A. Brooks, "Achieving artificial intelligence through building robots". MIT AI Memo 899, 1986.
- [Burt et al., 1989] P. Burt et al., *Proc. IEEE Workshop on Visual Motion*, 1989.
- [Feldman, 1985] J.A. Feldman, "Four frames suffice: A provisional model of vision and space", *Behavioral and Brain Sciences*, 1985.
- [Geman and Geman, 1984] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images", *IEEE Trans. PAMI* 6, 721-741, 1984.
- [Gibson, 1950] J.J. Gibson, *The Perception of the Visual World*, Houghton-Mifflin, Boston, 1950.
- [Horn, 1986] B.K.P. Horn, *Robot Vision*, M.I.T. Press, Cambridge, MA, 1986.
- [Horn and Weldon, 1988] B.K.P. Horn and E.J. Weldon, Jr., "Direct methods for recovering motion", *Int'l J. Comp. Vision* 2, 1988, 51-76.
- [IEEE, 1989] See for example the papers by V. Torre, A. Verri and F. Girosi in *Proc. IEEE Workshop on Visual Motion*, Irvine, CA, 1989.
- [Julesz, 1982] B. Julesz, "Visual pattern discrimination", *IRE Transactions on Information Theory* 8, 1962, 84-92.
- [Land and McCann, 1971] E.H. Land and McCann, J.J. "Lightness and retinex theory", *J. Opt. Soc. Am.* 61, 1-11, 1971.
- [LeClerc and Fua, 1987] Y. LeClerc and P. Fua, "Finding object boundaries using guided gradient ascent", in *Proc. Image Understanding Workshop*, 888-889, 1987.
- [Marroquin, 1985] J. Marroquin, "Probabilistic solution of inverse problems", Ph.D. Thesis, M.I.T., 1985.
- [Marr, 1982] D. Marr, *Vision*, W.H. Freeman: San Francisco, 1982.
- [Minsky, 1988] M. Minsky, *The Society of Mind*, 1988.
- [Mumford and Shah, 1985] D. Mumford and J. Shah, "Boundary detection by minimizing functionals", *Proc. IEEE CVPR*, 22-25, 1985.
- [Negahdaripour, 1986] S. Negahdaripour, Ph.D. Thesis, M.I.T., 1986.
- [Nelson] R. Nelson, "Detecting moving objects", personal communication (Technical Report, Dept. of Computer Science, University of Rochester).
- [Nelson, 1988] R. Nelson, "Visual homing using an associative memory", Ph.D. Thesis, University of Maryland, 1988.
- [Nelson and Aloimonos, 1989] R. Nelson and J. Aloimonos, "Using flow field divergence for obstacle avoidance in visual navigation", *IEEE Trans. PAMI* 11, 1102-1106, 1989.
- [Poggio and Edelman, 1989] T. Poggio and S. Edelman, *Proc. Image Understanding Workshop*, 1989.
- [Poggio et al., 1985] T. Poggio, V. Torre and C. Koch, "Computational vision and regularization theory", *Nature* 317, 214-319, 1985.
- [Regan] D. Regan, personal communication.
- [Ramachandran, 1989] V.S. Ramachandran, Invited talk, IEEE Workshop on Visual Motion, Irvine, CA, 1989.
- [Shulman and Aloimonos, 1988] D. Shulman and J. Aloimonos, "Boundary Preserving Regularization: Theory Part I", CAR-TR-356, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, 1988.
- [Spetsakis and Aloimonos, 1988] M.E. Spetsakis and J. Aloimonos, "Optimal computing of structure from motion using point correspondences in two frames", *Proc. ICCV*, 1988.
- [Weinshall] D. Weinshall, personal communication.
- [Zucker, 1988] S. Zucker, "The emerging paradigm of computational vision", *Ann. Rev. Comput. Sci.* 2, 69-89, 1987.

Efficient Parallel Processing in High Level Vision

Craig Reinhart *and Ramakant Nevatia †

Institute for Robotics and Intelligent Systems

Departments of Electrical Engineering and Computer Science

University of Southern California

Los Angeles, California 90089-0273

Abstract

We describe a methodology for developing efficient parallel implementations of high level vision algorithms. Efficiency is defined in terms of algorithm speedup, processor efficiency, system complexity, and programmer burden. Algorithm speedup and processor efficiency are critical issues in the parallel implementation of high level vision tasks as the required algorithms often utilize computationally intensive techniques. Furthermore, due to their usage of complex code and data structures, system complexity and maintenance costs can become excessive if care is not taken in the design of the implementation. Most researchers emphasize speedup and efficiency with little regard to system complexity and programmer burden. We show that through our design procedure, all four issues can be sufficiently addressed.

1 Introduction

Computer vision systems are comprised of tasks that can be categorized into three levels, low, mid, and high. Across the levels, a wide variety of algorithmic techniques are utilized ranging in complexity from simple repetitive processing to elaborate rule-based control structures. Also, the amount of active data at any given point in the system execution can range from tens of thousands of individual scalar values to a few multi-field record structures. Each diverse algorithm utilized in a computer vision system taxes a classical *von Neumann* (serial) architecture in one way or another.

In low-level vision, the multiplications and additions required by convolutional processing are easily executed on a serial machine but the amount of data on which they must operate (the image plane) overwhelm it. Conversely, the small number of abstract data structures utilized in high-level vision are easily maintained

by a *von Neumann* machine but the amount of processing and the complex control structures required to search through a solution space containing permutations of the data soon exceed the capabilities of the machine. To summarize, computer vision systems challenge serial machines through both data intensive and compute intensive operations. These challenges have made parallel implementation of computer vision systems an important topic within the computer vision research community [Ahuja and Swamy, 1984, Weems and Levitan, 1987, Kuehn *et al.*, 1985, Little *et al.*, 1987, Rosenfeld *et al.*, 1986, Hamey *et al.*, 1988, Stout, 1988, Sunwoo and Aggarwal, 1989].

We are interested in investigating the inherent complexities of computer vision systems and how those complexities can be tolerated via *efficient* use of a parallel processor architecture. We define *efficient* in terms of four measures.

Algorithm Speedup is a measure of the reduction in execution time when moving from a sequential to a parallel algorithm implementation. This is a standard measure in the study of parallel processing.

Processor Efficiency, also referred to as load balancing, is a measure of the amount of inherent parallelism, or conversely, the amount of inherent serialism, within an algorithm as well as how well suited the target parallel processor architecture is to the algorithmic requirements. This too is a standard measure in the study of parallel processing.

System Complexity is a measure of how closely the parallel implementation of the algorithm resembles the serial implementation, or conversely, how closely it resembles the parallel processor architecture. This is a measure that we are introducing as it plays an important role in the life cycle of a computer system, both software and hardware.

Programmer Burden is a measure of the degree of difficulty in developing and maintaining the parallel algorithm implementation. This is also a measure that we are introducing as it too plays an important role in the life cycle of a computer system.

An abundance of research into the parallel implementation of low and mid-level vision tasks on a variety of machines has been performed [Rice and Jamieson, 1985, Little *et al.*, 1987, Kuehn *et al.*, 1985, Stout, 1988, Levitan, 1984, Weems, 1988, Hamey *et al.*, 1988] but lit-

*Supported by the Hughes Aircraft Company Fellowship Program

†This research was supported by the Defense Advanced Research Projects Agency, monitored by the Air Force Wright Aeronautical Laboratories under contract F33615-87-C-1436

tle has been done with respect to high-level vision. Furthermore, researchers have placed dramatic emphasis on the issues of algorithm speedup and processor efficiency, especially speedup, with little or no regard to system complexity and programmer burden. Typically, the derived parallel implementations provide good measures of speedup and efficiency at the cost of obscure software and costly, custom built hardware.

In our approach, rather than select a parallel architecture then map an algorithm onto it, as is usually done, we perform some basic analysis steps in order to identify the inherent parallelism contained within the algorithm. We then specify the components of a parallel processor architecture that is well suited to the requirements of the algorithm. For a complete computer vision system comprised of a variety of algorithms, we specify an architecture for each algorithm that is well suited to that algorithm. These architectures can then be realized by either a single heterogeneous or reconfigurable parallel processor architecture. Through this approach we are able to address the issues of system complexity and programmer burden as well as algorithm speedup and processor efficiency.

Due to the need for increased through-put in high-level vision algorithms and the lack of research towards this end as well as the abundance of results available in the parallel implementation of low and mid-level vision algorithms, our studies are centered around high-level vision. In applying our approach to the parallel implementation of a relaxation based image matching algorithm [Medioni and Nevatia, 1984] we were able to:

- Achieve *significant* algorithm speedup.
- Achieve *significant* processor efficiency.
- Design a parallel processor architecture consisting of commercially available components.
- Utilize software that is nearly identical to that used in the serial implementation.

In the following sections we present our methodology for developing parallel implementations of computer vision algorithms and the application of the methodology to the relaxation based image matching algorithm.

2 The Methodology

Research into the parallel implementation of computer vision systems classically begins with the specification of a parallel processor architecture [Little *et al.*, 1987, Stout, 1988, Reisis and Prasanna-Kumar, 1987]. This includes the specification of various organizational parameters such as: *Programming model*, SIMD, MIMD, or MISD [Flynn, 1972], *Processing elements* (PEs), simple or complex instruction set; *Processing element coupling*, tightly (shared memory) or loosely (message passing); *Processor homogeneity*, homogeneous (identical processing elements) or heterogeneous (two or more different types of processing elements), *Processor synchronization*, synchronous, asynchronous, or loosely synchronous, and *Communication network topology*, cube, mesh, pyramid, ... Details on these and other organizational parameters can be found in [Hwang and Briggs, 1984].

Once a parallel processor architecture has been designed and an algorithm selected, one then proceeds to implement the algorithm on the architecture. This is a two step process. The first step is called the *mapping problem* [Bokhari, 1981] and involves two steps of its own. The second step is development of the actual code. We will not discuss the coding step as it involves the same effort as for a serial algorithm once the mapping problem has been solved.

The mapping problem is solved in two steps, the first involves *partitioning* the algorithm into independent processes and the second, *assigning* the processes to individual processing elements. A formal statement of the problem is: *the search for a correspondence between the interaction pattern of the algorithm processes and the communication network topology of the architecture*. A good solution, or mapping, is one that minimizes the communication overhead and thus maximizes the efficiency and the speedup.

With this approach, if an algorithm is not well suited to the given architecture, the designer is forced into developing an obscure algorithm implementation which resembles the architecture more so than the original algorithm specification.

In our methodology we approach the problem from the opposite direction. That is, we begin by analyzing the algorithm to determine its processing requirements then, using these findings, we specify a parallel processor organization that is well suited to the requirements. We proceed in four basic steps:

• Control Structure Analysis

In this step we identify the independent processes that constitute the algorithm through inspection of the processing constructs. Of primary interest are iterative constructs (loops) that determine the overall complexity of the algorithm and offer potential for parallelization. This step results in the identification of the inherent parallelism contained within the algorithm.

• Data Structure Analysis

In this step we determine the data requirements of each process identified above. The result of this step is the specification of which data structures to partition and how to partition them (distribute them among processes.)

• Communication Analysis

Identification of the independent processes and the data structure partitioning scheme will determine the communication requirements between the processes. That is, a data structure may be distributed among processes such that one process is assigned a data item required by another process to complete its task. In this step such requirements are determined as well as the appropriate communication protocols for their implementation, such as synchronous message passing among all processes, asynchronous message exchanges between two processes, message broadcasting and reduction. The result of this process will lead to the specification of

the communication network topology of the architecture.

• Architecture Specification

Given the results of the previous steps, this step is where we specify the architecture in terms of its organizational parameters. The result is the specification of a parallel processor architecture well suited to the requirements of the specified algorithm in terms of speedup, efficiency, system complexity, and programmer burden.

We have found that this approach produces high degrees of speedup and efficiency via software that resembles the serial implementation of the algorithm and is therefore no more difficult to develop and maintain. Furthermore, this approach lends itself to the design of parallel implementations of complete computer vision systems (heterogeneous algorithm suites) which can be implemented via a reconfigurable or a heterogeneous parallel processor architecture.

3 Image Matching - An Application

3.1 Overview

Matching of two images (or a map and an image) is a fundamental operation in computer vision. Various solutions to the problem of finding correspondences between images have been proposed ranging from correlation [Rosenfeld and Kak, 1976] to graph isomorphism [Ghahraman *et al.*, 1980]. One primary distinction among the proposed solutions is the level of description at which the matching is performed. Correlation based techniques typically operate directly on sensor data (pixels) whereas graph based approaches often utilize semantic structures such as roads and buildings.

The image matching algorithm used in our study utilizes a discrete relaxation based approach to matching. It determines correspondences between line segments detected in each image based on symbolic descriptions of the segments as well as geometrical relationships between segments. The algorithm iterates over the solution space until a stable state is converged upon.

This algorithm was selected for study due, primarily, to its applicability to high-level vision. But, the basic approach utilized in the algorithm, relaxation, has been used in other applications as well [Waltz, 1972, Rosenfeld *et al.*, 1976, Faugeras and Price, 1981, Rosenfeld and Smith, 1981, Terzopoulos, 1986, Rutkowski *et al.*, 1981]. Therefore, the results of this study can be generalized to various other applications in low, mid, and high-level vision.

In the following sections we present details of the application of our methodology to the relaxation based image matching algorithm. We present a brief description of the algorithm, application of the four steps that constitute our methodology, and a discussion of the results of the application in terms of our four measures, algorithm speedup, processor efficiency, system complexity, and programmer burden.

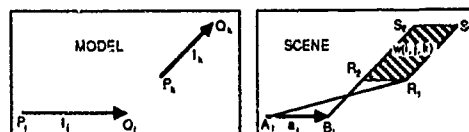


Figure 1: Window construction.

3.2 Algorithm Description

This image matching algorithm [Medioni and Nevatia, 1984] receives input images from two independent sources and then attempts to construct a list of correspondences between them using a relaxation based approach. We provide an overview of the algorithm with enough detail to discuss our algorithm mapping methodology. For details and explanations beyond the scope of our discussion, the reader should see the referenced work.

The primitives used by the image matching algorithm are *linear segments*, represented symbolically by their end point coordinates, orientation, and average contrast. Given two sets of linear segments extracted from two images (or an image and a map), the object is to find correspondences between the segments of each set based on their symbolic descriptions (local constraints) and on the geometrical relationships between segments of the same image (global constraints.) The assumptions made prior to matching are that: 1) the orientations of the two images are nearly the same; and 2) the scaling factor from one image to the other is approximately known.

The set of primitives, $A = \{a_i | 1 \leq i \leq n\}$, from one image is called the *SCENE* and the primitives, a_i , are called *OBJECTS*. The set of primitives, $L = \{l_j | 1 \leq j \leq m\}$, from the other image is called the *MODEL* and the primitives, l_j , are called *LABELS*. The algorithm proceeds to compute the quantity $p(i, j)$ in $\{0, 1\}$, which is the *POSSIBILITY* that object a_i corresponds to label l_j . It is possible that an object has no corresponding label due to occlusion or scene change, that several objects correspond to the same label due to fragmentation, or that an object corresponds to several labels due to merging. The method for computing $p(i, j)$ relies on geometrical constraints, that is, when a label, l_j , is assigned to an object, a_i , we expect to find an object, a_h , with a label, l_k , in an area defined by i, j , and k . The area is called a *WINDOW* and is denoted $w(i, j, k)$.

The method for computing $w(i, j, k)$ is as follows. The object, a_i , is represented by the two dimensional vector $A_i B_i$ and the label, l_j , by $P_j Q_j$. By "sliding" l_j over a_i , an area is described by the corresponding motion of label l_k , $P_k Q_k$ (figure 1.) This parallelogram shaped area is the window $w(i, j, k)$.

Two object/label assignments, (i, j) and (h, k) , are *COMPATIBLE*, $(i, j)C(h, k)$, if and only if object a_h lies within window $w(i, j, k)$ and object a_i lies within window $w(h, k, j)$.

Using these definitions, the algorithm searches for object/label correspondences by first identifying all possible correspondences based on the symbolic descriptions of the objects and labels. This set of correspondences

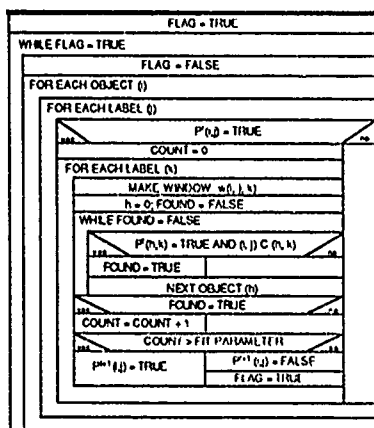


Figure 2: Image matching algorithm primary flow.

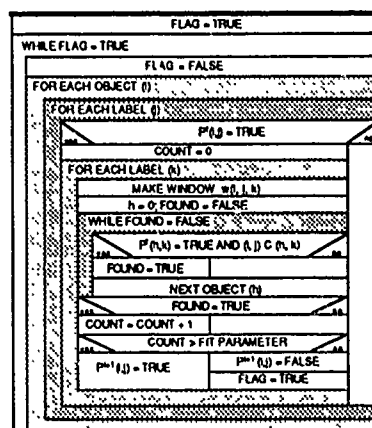


Figure 4: Image matching primary control loops.

```

flag = 1;
while (flag) {
    flag = 0;
    /* Iteration of possibilities/compatibilities. */
    for (i = 0; i < number_of_objects; ++i)
        for (j = 0; j < number_of_labels; ++j)
            if (p[j][i]) { /* if P(i,j) == 1 */
                card_s = 0;

                /* Compute the degree of support for */
                /* the object/label assignment. */

                for (k = 0; k < number_of_labels; ++k) { /* for all labels */
                    make_window(objects[i], labels[j], labels[k], win_ijk);
                    h = 0; found = 0;
                    while ((h < number_of_objects) && (!found)) {
                        if (p[k][h] && in_window(objects[h], win_ijk))
                            found = compatible(objects[i], labels[j], objects[h], labels[k]);
                        ++h;
                    } /* while ((h < number_of_objects) ... */
                    if (found)
                        ++card_s;
                } /* for (k = ... */

                if (card_s < q) {
                    flag = 1;
                    p[j][i] = 0;
                } /* if (card_s ... */
            } /* if (p[j][i] ... */
        } /* while (flag) ... */
}

```

Figure 3: Serial code for image matching algorithm.

constitutes the possibilities at iteration step 0, $p^0(i, j)$. Subsequent values of $p(i, j)$ are computed by the *iteration formula*:

$$\begin{aligned} & \forall (i, j), p^{t+1}(i, j) = 1 \text{ if } p^t(i, j) = 1 \text{ AND} \\ & \exists \text{ subset } S \text{ of } [1, m] \text{ (labels) with } q \text{ elements such that} \\ & \forall s \text{ in } S, \exists k \text{ in } [1, n] \text{ (objects) such that } p^t(k, s) = 1 \text{ and} \\ & \quad (i, j) C(k, s). \end{aligned}$$

3.3 Control Structure Analysis

In analyzing the control structure of an algorithm our objective is to determine its overall time complexity and to identify the specific structures that dictate that time complexity, typically loop constructs. We call these constructs *primary control structures*. Identification of the primary control structures will help us to identify independent processes and thus, identify areas where parallelism can be applied providing significant algorithm speedup.

The time complexity of the image matching algorithm is determined as follows. Given a scene containing n objects and a model containing m labels, the maximum number of possible object/label pairs is nm , which occurs when every object is similar to every label. At each iteration at most one object/label pair is discarded, that is, its possibility is set to 0, therefore, the process converges in at most nm iterations. During each iteration the algorithm computes the possibility of the object/label pair which is a measure of how well it 'fits' with the remaining object/label pairs. In the worst case, this requires investigating nm pairs. Therefore, the complexity of the algorithm is $O(n^2m^2)$. If we assume an equal number of objects and labels, m , the algorithm time complexity can be expressed as $O(m^4)$. Figure 4 shows, pictorially, the four nested loops which implement this time complexity. These constitute the primary control structures.

Nested within the four loops is the *possibility computation*. As described above, it consists of checking whether or not a given object/label pair has any *compatible* object/label pairs. This, in turn, requires the computation of a window and the search for an object within it. Once a candidate object/label pair, (a, l_j) , has been queued, the possibility computation, $p(i, j)$, can proceed as m^2 independent computations. Each computation is structured so that it operates on an isolated data set, that is, successive passes through the inner loops (the possibility computation) are independent of one another. Thus, the possibility computation can proceed as multiple parallel processes and has the potential to provide significant al-

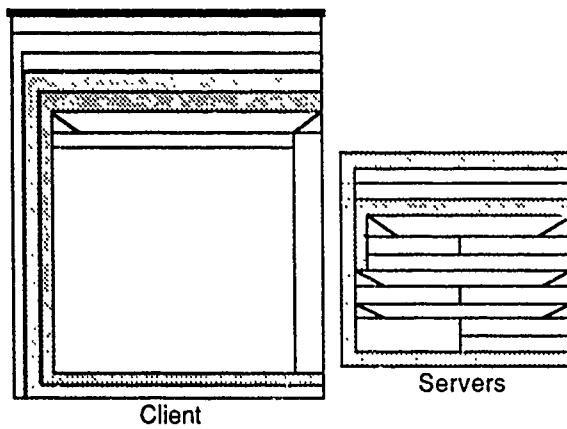


Figure 5: Client/Server algorithm partitioning.

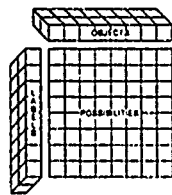


Figure 6: Image matching primary data structures.

algorithm speedup. For these reasons, it constitutes our process partitioning scheme.

Having selected the possibility computation as the process with which to partition the algorithm, we have produced a client/server model. That is, one process will queue possible object/label pairings via the outer two primary control loops, constituting the client, and a set of independent processes will determine the possibility of that pairing via execution of the inner two control loops and their encompassed procedures in a distributed fashion, constituting the servers. Figure 5 shows the client/server algorithm partitioning.

3.4 Data Structure Analysis

Having identified the possibility computation as the task on which to partition the algorithm into processes, we must now determine the data requirements of each computation. In doing so we will identify the *primary data structures* and determine an appropriate partitioning of these structures.

For the image matching algorithm, three primary data structures can be identified. The first two are linear arrays of size m of symbolic records, one array each for storage of the set of objects and the set of labels. The third data structure is an $m \times m$ matrix of logical values that store the results of the possibility computation, $p^t(i, j)$, for each iteration, t . Figure 6 shows the primary data structures, pictorially.

Each possibility computation (process) requires two entries from the object array, a_i and a_h , and two entries from the label array, l_j and l_k . All processes receive

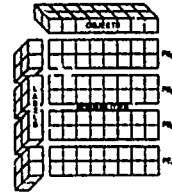


Figure 7: Image matching horizontal swath partitions.

the same (a_i, l_j) pair, the object/label assignment under consideration, and each receives a unique (a_h, l_k) pair, an object/label assignment that determines the global consistency of the pair under consideration. From these inputs the windows, $w(i, j, k)$ and $w(h, k, j)$, are formed. The relation $(i, j)C(h, k)$ is then computed by determining whether or not a_h lies within $w(i, j, k)$ and a_i lies within $w(h, k, j)$. A value of 1 is returned if the relation holds, otherwise a value of 0 is returned. The value of $p^{t+1}(i, j)$ is determined by summing the results from all of the individual processes and comparing that sum to the fit parameter, q .

If we assume the availability of $N = m^2$ processing elements, the obvious way of partitioning the data structures is to assign each PE, $0 \leq p \leq N-1$, an object/label pair, $(a_h, l_k) \in A \times L$. If the number of processing elements available is less than m^2 , that is, $N \ll m^2$, then the most intuitive way, from a programmer's viewpoint, to partition the data structures is to assign each PE, $0 \leq p \leq N-1$, to a $1/N$ sized portion of the label array and the entire object array thus giving each a set $S_p = \{(a_h, l_k) | 1 \leq h \leq m, p * (m/N) \leq k \leq p * (m/N) + m/N - 1\} \forall p : 0 \leq p \leq N-1$ of objects and labels. This creates N horizontal swaths through the possibility matrix as depicted in figure 7. These horizontal swaths constitute our data partitioning scheme.

3.5 Communication Analysis

Having designed our process and data partitions, we must now identify the inter-process communication required to complete the parallel implementation.

As described previously, a possibility computation requires access to the object/label pair under consideration, (a_i, l_j) , provided by the client process, and the set of possible object/label pairs from which the server processes compute a degree of support. The set of possible pairs are statically distributed among the server processes once, upon algorithm initiation, as described above. Conversely, the pair (a_i, l_j) must be provided to each server, dynamically, by the client process. This is achieved via a *broadcast* operation from the client to every server.

Having received (a_i, l_j) , each server process computes a degree of support for the pair based on its set of possible object/label pairs (its data partition.) Upon completion, each server reports its degree of support to the client where the individual degrees of support are combined into a single result and the possibility computation, $p^t(i, j)$, is completed. This is achieved via a *reduction* operation from every server to the client.

Finally, the client must report $p^i(i, j)$ to the server process whose data partition includes the pair (a_i, l_j) so that it can update its possibility value. This is achieved via a point-to-point *send/receive* operation from the client to the particular server.

In summary, our process/data partitioning scheme requires three types of communication: 1) broadcast; 2) reduction; and 3) point-to-point send/receive.

This concludes the analysis steps of our methodology as applied to the image matching algorithm. We have described the algorithm, identified its primary control structures, identified its primary data structures, partitioned it into independent processes, and identified all required inter-process communication. Our remaining task is to specify a parallel architecture well suited to the requirements identified by our analysis. This is presented in the following section. We then present an evaluation of the system design arrived at via our methodology through architecture simulation and actual implementation.

3.6 Architecture Specification

In specifying a parallel processor architecture we must address various organizational parameters: *Programming model*; *Processing element type*; *Processing element coupling*; *Processor homogeneity*; *Processor synchronization*; and *Communication network topology*. Whereas in the classical approach this is done prior to the algorithm analysis, that is, the parallel implementation of the algorithm is specified for a particular parallel architecture, we base our specification of these parameters on the results of our algorithm analysis. In the following paragraphs we address each of these organizational parameters and discuss how they are influenced by the processing requirements of the image matching algorithm.

Programming Model. The image matching algorithm (more specifically, the possibility computation) contains various processing steps that are data dependent, that is, all data items are not processed identically. The *Multiple Instruction Multiple Data* programming model is best suited to this situation. In this model each processing element can execute code dictated by its particular data items. Conversely, the algorithm could be implemented under the *Single Instruction Multiple Data* programming model, as demonstrated in [Reisis and Prasanna-Kumar, 1987], but processing elements would spend a great deal of time "idling" through code which is not applicable to their data items and thus, reduce the processor efficiency.

Processing Element Type. Computation of the compatibility relationship, $(i, j)C(h, k)$, between to pairs of object/label correspondences, (a_i, l_j) and (a_h, l_k) , requires computation of two windows, $w(i, j, k)$ and $w(h, k, j)$, as well as whether or not the objects a_i and a_h lie within the respective windows. These computations require the use of transcendental functions as well as floating point arithmetic (unless integerization is performed.) Therefore, the processor utilized must support these computations. Furthermore, to reduce system complexity and programmer burden, the processor must be programmable in a high-level language that allows

specification of the primary data structures in a natural way, that is, via multi-field records. Processors best suited to these constraints are of the complex instruction set variety such as a general purpose microprocessor.

Processing Element Coupling. As the communication between processes is in bursts, that is, at the beginning of each possibility computation (the broadcast) and at the end of each possibility computation (the reduction), a tightly coupled or shared memory system would not suffice because of memory access conflicts. Without special protocols to allow concurrent reading and writing of memory, a communication bottle neck would exist. Better suited to the algorithm is a loosely coupled or message passing architecture. These systems facilitate high bandwidth communication without the requirement of special purpose hardware.

Processor Homogeneity. Our partitioning scheme provides each server process with identical tasks. The client process is computationally similar to the server processes in that it utilizes the same data structures as well as similar logic. Therefore, the parallel architecture should be homogeneous, that is, comprised of a set of identical processing elements. This facilitates programming (reduction of programmer burden) as well as hardware interfacing of processing elements (reduction of system complexity.)

Processor Synchronization. In light of the fact that there is computational similarity between all of the identified processes as well as data dependent processing, the parallel architecture should operate in loosely synchronous mode. That is, all processes incorporate identical copies of the program, with the exception of the client process, and execute under control of their own program counter. Synchronization occurs only at points of communication. As we shall see, this also facilitates programmability of the implementation which reduces system complexity and programmer burden.

Communication Network Topology. Perhaps the most interesting aspect of a parallel processor architecture is its communication network topology, the processing element interconnect pattern. As we showed via the communication analysis, the image matching algorithm places three constraints on the communication network topology. The first is that it must facilitate an efficient broadcast operation, the second is that it must facilitate an efficient reduction operation, and the third is that it must facilitate an efficient point-to-point send/receive operation. In the following paragraphs we consider each of these constraints.

With regard to the broadcast operation, the ideal message passing architecture is one containing a single common bus to which all processing elements are connected. In this topology a broadcast operation is completed in $O(1)$ time.

With regard to the reduction operation, the ideal algorithm requires $\Omega(\log n)$ time, that is, "order no less than $\log n$ time", assuming concurrent read and write operations are forbidden [Cole and Vishkin, 1986]. This ideal time is achieved by an algorithm that utilizes a divide and conquer approach. The result is obtained by dividing the data set into two halves, finding the two partial

results, and combining the partial results to get the final result. The dividing is done recursively until the data sets are indivisible. Such a divide and conquer scheme yields a binary tree with n^2 nodes with the data items starting at the leaves. For the image matching algorithm the data items, the objects and labels used to determine the global validity of a queued object/label correspondence, can be distributed among all nodes of the tree, not just the leaves.

With regard to the point-to-point send/receive operation, the ideal message passing architecture is, again, one containing a single common bus to which all processing elements are connected. In this topology a send/receive operation is completed in $O(1)$ time.

The reduction operation produces the most stringent constraint dictated by the image matching algorithm. A communication network topology that facilitates this operation will also facilitate the other two as they are of lower order complexity. Therefore, for parallel implementation of the image matching algorithm, the processing elements should be connected via a binary tree topology.

To summarize, the organizational parameters of a parallel processor architecture that is well suited to the image matching algorithm should be specified as follows:

- Programming model – MIMD
- Processing elements – Complex Instruction Set Computers
- Processor coupling – Loosely Coupled
- Processor homogeneity – Homogeneous
- Processor synchronization – Loosely Synchronous
- Communication network topology – Binary Tree

This completes the application of our methodology to the image matching algorithm. In the next section we present an evaluation of the system design in terms of our measures; algorithm speedup, processor efficiency, system complexity, and programmer burden.

3.7 System Evaluation

Having completed our parallel implementation of the image matching algorithm, we now present an evaluation of the implementation in terms of our four measures. The evaluation is performed on the basis of three "data points." First, we use a serial implementation of the algorithm as a baseline with which comparisons can be performed. Second, we use a simulation developed to analyze the implementation relative to any number of processing elements. Third, we use an actual system implementation utilizing INMOS Transputers [INM, 1989] to bring validity and feasibility to the entire study.

As we stated earlier, most research in the field of parallel processing of computer vision algorithms is primarily concerned with algorithm speedup and processor efficiency. For this reason we begin our evaluation and discussion with these two measures.

Algorithm speedup is defined as the ratio of elapsed time when executing a program on a single processor to the elapsed time when N processors are available.

That is, for N processing elements, algorithm speedup is defined as

$$S_N = \frac{T_1}{T_N}$$

where T_1 and T_N are the elapsed times for 1 and N processing elements, respectively.

Processor efficiency is defined as the average utilization of the available processing elements and can be specified in terms of algorithm speedup, S_N . For N processing elements, processor efficiency is defined as

$$E_N = \frac{S_N}{N}$$

If the efficiency, E_N , of a parallel implementation remains constant (ideally 1) as the number of processing elements, N , is increased, the parallel implementation of the algorithm is said to have achieved *linear speedup*.

3.7.1 Complexity Analysis

Previously we determined the complexity of the image matching algorithm to be $O(m^4)$, assuming an equal number of objects and labels, m . This is due to the nested loop structure of the algorithm where every object/label pair, (a_i, l_j) , is checked against every other object/label pair, (a_h, l_k) for compatibility.

In our partitioning strategy, we distribute the m^2 compatibility computations for each object/label pair possibility computation evenly among the N processing elements. Therefore, barring the existence of any data dependencies or overhead, we expect to achieve $O(N)$ speedup and complete processor utilization, that is, an efficiency of 1. Unfortunately, both data dependencies and overhead exist.

The data dependencies contained within the image matching algorithm can be expressed in terms of the possible correspondences between objects and labels. Let us define the value P_j to be the set of possible object correspondences for each label l_j , $1 \leq j \leq m$. We can then define

$$\begin{aligned} \hat{P} &= \max_{j=1}^m |P_j|, \\ \bar{P} &= \frac{\sum_{j=1}^m |P_j|}{m}, \text{ and} \\ k &= \frac{\hat{P}}{\bar{P}}. \end{aligned}$$

The value k is an indication of how evenly the object/label correspondences are distributed. For instance, if every label forms possible correspondences with the same number of objects, k will be 1. Conversely, if one label forms possible correspondences with a large number of objects and the remaining labels form possible correspondences with a small number of objects, then k will be large.

Using these definitions, the *expected* values for algorithm speedup and processor efficiency for our implementation of the image matching algorithm (barring any overhead) are

$$\begin{aligned} S_N^e &= \frac{N}{k} \text{ and} \\ E_N^e &= \frac{1}{k}. \end{aligned}$$

As an appeal to one's intuition, consider the following cases. Recall that our data partitioning scheme calls for the assignment of a set of object/label pairs to each processing element. If all sets contain an equal number of possible correspondences, then

$$\hat{P} = \bar{P} \Rightarrow k = 1 \Rightarrow \\ S_N^e = N \text{ and } E_N^e = 1.$$

This implies that each processing element is assigned the same amount of work. If one set contains more possible correspondences than all of the rest, $\exists j: |P_j| \gg |P_i| \forall i \neq j, 1 \leq i \leq m$, then

$$\hat{P} \gg \bar{P} \Rightarrow k \gg 1 \Rightarrow \\ S_N^e \ll N \text{ and } E_N^e \ll 1.$$

This implies that the processing element assigned the set P_j must do more work than any of the other processing elements.

These expected values are to be considered estimates of the overhead incurred by the implementation due to data dependencies. One must remember that the actual distribution of possible object/label correspondences is dynamic as it is the goal of the algorithm to reduce this to a canonical set of correspondences via the relaxation operation. Furthermore, the algorithm contains some inherently serial operations, those of the client process, that must be taken into consideration in light of the overall performance analysis. Actual values of algorithm speedup and processor efficiency will vary due to this dynamic behavior and serialism. The goal is to minimize the effects of these on the performance of the parallel implementation.

3.7.2 Measured Performance

To measure the actual values of algorithm speedup and processor efficiency we devised three test cases. The first is comprised of two identical images containing multiple vertical lines. In this scenario $k = 1$. The second is comprised of an image containing one vertical line and multiple horizontal lines and an image containing one horizontal line and multiple vertical lines. In this scenario $k = m/2$ where m is the number of labels. The third is comprised of two identical images containing lines extracted from an airfield image. In this scenario $k = 1.67$.

Table 1 shows the execution times for the three scenarios when instantiated with various problem sizes. The first four rows are for the first scenario with the number of labels, m , being 12, 24, 36, and 48. The next three rows are for the second scenario with the number of labels being 50, 100, and 200. The last three rows are for the third scenario with the number of labels being 51, 102, and 153. Simulation runs were done with the number of processing elements being 1, 2, 3, 4, 15, and m , the number of labels. These are represented by the six columns.

Table 2 shows the measured speedup for each of the test cases and table 3 shows the efficiency. One should note that these measured values do not include overhead for inter-processor communication. They strictly reflect the algorithm speedup and processor efficiency as affected by our process and data partitioning schemes and the data dependencies.

To observe the effects of inter-processor communication on the overall performance (as well as to demonstrate a complete application of our methodology) we also developed an actual parallel processor system based on our implementation of the image matching algorithm.

Problem	T_1	T_2	T_3	T_4	T_{15}	T_m
12	10.53	5.22	3.73	3.13	—	1.01
24	167.57	97.08	63.23	53.43	18.83	9.31
36	870.90	473.82	342.62	271.02	90.52	31.11
48	2743.32	1487.00	1057.93	792.52	283.43	72.71
50	6.43	3.45	2.37	1.78	0.58	0.21
100	33.60	18.10	12.13	9.15	2.75	0.51
200	271.80	140.72	95.78	68.75	19.15	1.81
51	75.07	39.37	28.67	21.78	6.82	2.11
102	1091.93	552.42	375.27	300.35	83.28	18.31
153	6620.68	3396.95	2267.65	1700.68	547.67	78.01

Table 1: Execution times from simulation.

Problem	S_1	S_2	S_3	S_4	S_{15}	S_m
12	1.00	2.02	2.82	3.36	—	9.75
24	1.00	1.73	2.65	3.14	8.90	18.02
36	1.00	1.84	2.54	3.21	9.62	27.96
48	1.00	1.84	2.59	3.46	9.68	37.73
50	1.00	1.86	2.71	3.61	11.09	27.96
100	1.00	1.86	2.77	3.67	12.22	58.95
200	1.00	1.93	2.84	3.95	14.19	145.35
51	1.00	1.91	2.62	3.45	11.01	34.59
102	1.00	1.98	2.91	3.64	13.11	59.67
153	1.00	1.95	2.92	3.89	12.09	84.86

Table 2: Speedup from simulation.

The system is comprised of one to four INMOS Transputers but is expandable to incorporate any number of processing elements without any system redesign.

Tables 4, 5, and 6 show the measured execution times, algorithm speedup, and processor efficiency, respectively, for the various test cases and problem sizes.

Although our data points are sparse, the tables do indicate the following trends in terms of algorithm speedup and processor efficiency for our parallel implementation of the image matching algorithm:

- The implementation is most effective when the problem size is large, that is, when the number of possible object/label correspondences is large.
- The implementation is most effective when the number of processing elements is less than or equal to

Problem	E_1	E_2	E_3	E_4	E_{15}	E_m
12	1.00	1.00	0.94	0.84	—	0.81
24	1.00	0.86	0.88	0.78	0.59	0.75
36	1.00	0.92	0.85	0.80	0.64	0.78
48	1.00	0.92	0.86	0.87	0.65	0.79
50	1.00	0.93	0.90	0.90	0.74	0.55
100	1.00	0.93	0.92	0.92	0.81	0.58
200	1.00	0.97	0.95	0.99	0.95	0.73
51	1.00	0.95	0.87	0.86	0.73	0.68
102	1.00	0.99	0.97	0.91	0.87	0.58
153	1.00	0.97	0.97	0.97	0.81	0.55

Table 3: Efficiency from simulation.

Problem	T ₁	T ₂	T ₃	T ₄
12	8	5	4	3
24	119	71	50	40
36	581	338	238	185
48	1815	1060	747	582
50	7	5	4	4
100	31	23	19	18
200	159	106	83	78
51	60	34	23	21
102	839	439	303	233
153	4063	2112	1373	1080

Table 4: Execution times from Transputer implementation.

Problem	S ₁	S ₂	S ₃	S ₄
12	1.00	1.60	2.00	2.67
24	1.00	1.68	2.38	2.98
36	1.00	1.72	2.44	3.14
48	1.00	1.71	2.43	3.19
50	1.00	1.26	1.52	1.54
100	1.00	1.36	1.63	1.73
200	1.00	1.50	1.91	2.05
51	1.00	1.76	2.40	2.86
102	1.00	1.91	2.77	3.60
153	1.00	1.92	2.96	3.76

Table 5: Speedup from Transputer implementation.

the number of labels (when data dependencies are taken into account.)

- In light of the previous items, inter-processor communication does not dominate the implementation.

We now focus our attention on our two new measures, system complexity and programmer burden.

3.7.3 System Development and Maintenance

As stated earlier, system complexity is a measure of how closely the parallel implementation of the algorithm resembles the serial implementation. It can also be viewed as the amount of effort (cost) required to realize the parallel implementation of the algorithm.

Previously, we showed a program segment for the image matching algorithm's primary control structures as

Problem	E ₁	E ₂	E ₃	E ₄
12	1.00	0.80	0.67	0.67
24	1.00	0.84	0.79	0.76
36	1.00	0.86	0.81	0.79
48	1.00	0.86	0.81	0.86
50	1.00	0.63	0.51	0.39
100	1.00	0.68	0.54	0.43
200	1.00	0.75	0.64	0.51
51	1.00	0.88	0.80	0.72
102	1.00	0.96	0.92	0.90
153	1.00	0.96	0.99	0.94

Table 6: Efficiency from Transputer implementation.

```

flag = 1;
while (flag) {
    flag = 0;
    /* Iteration of possibilities/compatibilities. */
    for (i = 0; i < number_of_objects; ++i)
        for (j = 0; j < number_of_labels; ++j)
            if (p[i][j]) { /* if P(i,j) == 1 */
                card_s = 0;

                SEND OBJECT/LABEL ASSIGNMENT PAIR TO CHILDREN
                (BROADCAST).

                /* Compute the degree of support for the object/label */
                /* assignment provided by this PEs 1/nth of the label table. */
                for (k = 0; k < my_share; ++k) { for my share of labels */
                    make_window(objects[i], labels[j], labels[k], win_ijk);
                    h = 0; found = 0;
                    while ((h < number_of_objects) && (!found)) {
                        if (p[k][h] && in_window(objects[h], win_ijk))
                            found = compatible(objects[i], labels[j], objects[h], labels[k]);
                        ++h;
                    } /* while ((h < number_of_objects) ... */
                    if (found)
                        ++card_s;
                } /* for (k = ... */

                RECEIVE CONTRIBUTIONS FROM CHILDREN (REDUCTION).

                card_s += left_child.contribution;
                card_s += right_child.contribution;

                if (card_s < q) {
                    flag = 1;
                    p[i][j] = 0;
                } /* if (card_s ... */

                SEND CHANGES TO THE PE WITH PAIR (i,j).

            } /* if (p[i][j]) ... */
        } /* while (flag) ... */
}

```

Figure 8: Client code for parallel image matching algorithm.

implemented on a serial machine. The programming language was 'C'[Kernigan and Ritchie, 1978]. In figures 8 and 9 we show program segments for the client and server processes, respectively, also written in 'C'. Note that the algorithm-specific constructs are identical in the serial and parallel programs. The only differences are the inclusion of the subroutine calls to perform inter-processor communication. Therefore, one can conclude that the complexity of the parallel software is no greater than that of the serial implementation. This is attributable to the fact that the parallel implementation was designed based on the structure of the algorithm, and not on the structure of the parallel processor architecture.

Programmer burden is a measure of the degree of difficulty in developing and maintaining the parallel algorithm implementation. It can also be viewed as the amount of effort (cost) required to modify and debug the parallel software in light of algorithm modifications. In computer vision, this measure is critical due to the fact that the vision problem is far from being solved and algorithm refinements arrive at a high rate.

As discussed above, the software for the parallel implementation of the image matching algorithm is identical to that of the serial implementation as far as algorithm specific constructs are concerned. Therefore, algorithm debugging and modification can take place in the serial environment where advanced tools are readily available and then ported directly to the parallel environment. Using this technique, once we achieved a "bug free" version of the algorithm on a serial computer, we were able to

```

done = 0;
while (!done) {

    RECEIVE OBJECT/LABEL PAIR FROM PARENT (BROADCAST).

    SEND OBJECT/LABEL PAIR TO CHILDREN (BROADCAST).

    /* Compute the degree of support for the object/label */
    /* assignment provided by this PE's 1/nth of the label table. */

    contribution = 0;
    for (k = 0; k < my_share; ++k) { for my share of labels */
        make_window(objects_j, labels_j, labels[k], win_jk);
        h = 0; found = 0;
        while ((h < number_of_objects) && (!found)) {
            if (p[k][h] && in_window(objects[h], win_jk))
                found = compatible(objects[j], labels[j], objects[h], labels[k]);
            ++h;
        } /* while ((h < number_of_objects) ... */
        if (found)
            ++card_s;
        } /* for (k = ... */

    RECEIVE CONTRIBUTIONS FROM CHILDREN (REDUCTION).

    card_s += left_child.contribution;
    card_s += right_child.contribution;

    REPORT THE CONTRIBUTION TO THE PARENT (REDUCTION).

    RECEIVE CHANGES WHEN APPLICABLE.

    SEND CHANGES TO CHILDREN WHEN APPLICABLE.

    } /* while (flag) ... */

```

Figure 9: Server code for parallel image matching algorithm.

get it running in parallel in approximately twelve hours. The primary effort was in validating the inter-process communication. But, once validated, the code that implements the communication is functionally portable to other algorithms that utilize the same communication network topology and need not be validated again.

Therefore, we can conclude that we have minimized the measure of programmer burden in that the development and maintenance efforts for the parallel implementation were performed, predominately, in the serial environment.

This image matching algorithm was previously mapped onto a parallel processor architecture via the classical approach of specifying an architecture then mapping the algorithm onto it [Reisis and Prasanna-Kumar, 1987]. In that study the specified architecture was a 2D mesh connected SIMD architecture consisting of relatively simple processing elements, a parallel processor architecture well suited to low-level computer vision tasks. By using elaborate data partitioning and memory access schemes, an implementation that theoretically achieves linear speedup (in the absence of data dependencies) was developed. When data dependencies are considered, the implementation achieves the same measures of algorithm speedup and processor efficiency as we presented. The study was purely theoretical and implementation was not actually carried out. To actually implement the system would be extremely difficult due to the nature of the data partitioning scheme, which is retinotopic based to match the communication network topology of the architecture. Furthermore, one can show that the effects of the data dependencies on speedup and efficiency are exaggerated due to synchronous nature of

the SIMD machine. Finally, any modification of the implementation (algorithm) would require intimate knowledge of the algorithm, the architecture, and the implementation as is the case with most "classical" parallel algorithm implementations.

We have shown that these situations can be overcome by design (or selection) of a parallel processor architecture based on the processing and data requirements of the algorithm rather than specifying the algorithm implementation to meet the specifications of the parallel processor architecture.

4 Summary

We have described a methodology for mapping algorithms onto parallel processor architectures. Utilizing our methodology to analyze, simulate, and implement a commonly used algorithmic technique, relaxation, we exposed various characteristics common among high-level vision algorithms that must be considered when designing a parallel implementation if the goals of maximized algorithm speedup, maximized processor efficiency, minimized system complexity, and minimized programmer burden are to be achieved. These characteristics include: 1) the use of complex program logic; 2) the existence of subtle data dependencies; 3) the use of heterogeneous data structures; and 4) the dynamic nature of the data.

As shown in [Reisis and Prasanna-Kumar, 1987], when designing an implementation targeted for a specific parallel processor architecture these issues either cannot be sufficiently addressed or require extremely convoluted, unintuitive solutions which lead to an implementation which is difficult to develop and maintain. In applying our methodology we have shown that all of these issues can be addressed without sacrificing any of the goals.

We are currently applying the methodology to other stand-alone computer vision algorithms as well as to complete computer vision systems to determine its utility in specifying a reconfigurable or heterogeneous parallel processor architecture for implementation of such an algorithm suite. We are also investigating the usefulness (cost versus payoff) of dynamic data partitioning (load balancing) schemes in their application to high-level vision algorithm implementations.

References

- [Ahuja and Swamy, 1984] N. Ahuja and S. Swamy. Multiprocessor pyramid architectures for bottom-up image analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(4):463-475, July 1984.
- [Bokhari, 1981] S. H. Bokhari. On the mapping problem. *IEEE Transactions on Computers*, C-30(3):207-215, March 1981.
- [Cole and Vishkin, 1986] R. Cole and U. Vishkin. Approximate and exact parallel scheduling with applications to list tree and graph problems. In *Proceedings of the 27th Symposium on Foundations of Computer Science*, pages 478-491, 1986.

- [Faugeras and Price, 1981] O.D. Faugeras and K. Price. Semantic description of aerial images using stochastic labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(6):633-642, November 1981.
- [Flynn, 1972] M. J. Flynn. Some computer organizations and their effectiveness. *IEEE Transactions on Computers*, C-21(9):948-960, September 1972.
- [Ghahraman et al., 1980] D. E. Ghahraman, A. K. C. Wong, and T. Au. Graph monomorphism algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-10(4):189-196, April 1980.
- [Hamey et al., 1988] L. G. C. Hamey, J. A. Webb, and I-C. Wu. Low-level vision on Warp and the Apply programming model. In J. S. Kowalik, editor, *Parallel Computation and Computers for Artificial Intelligence*, pages 185-199. Kluwer Academic Publishers, 1988.
- [Hwang and Briggs, 1984] K. Hwang and F. A. Briggs. *Computer Architecture and Parallel Processing*. McGraw-Hill, New York, 1984.
- [INM, 1989] INMOS Corporation. *The Transputer Databook*, 1989.
- [Kernigan and Ritchie, 1978] B. W. Kernigan and D. M. Ritchie. *The C Programming Language*. Prentice Hall, New Jersey, 1978.
- [Kuehn et al., 1985] J. T. Kuehn, H. J. Siegel, D. L. Tuomenoksa, and G. B. Adams III. The use and design of PASM. In S. Levialdi, editor, *Integrated Technology for Parallel Image Processing*, pages 133-152. Academic Press, 1985.
- [Levitani, 1984] S. P. Levitan. *Parallel Algorithms and Architectures: A Programmer's Perspective*. PhD thesis, University of Massachusetts, 1984. Computer and Information Sciences.
- [Little et al., 1987] J. J. Little, G. Brelloch, and T. Cass. Parallel algorithms for computer vision on the Connection Machine. In *Proceedings of the DARPA Image Understanding Workshop*, pages 628-638, February 1987.
- [Medioni and Nevatia, 1984] G. Medioni and R. Nevatia. Matching images using linear features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):675-685, November 1984.
- [Reisis and Prasanna-Kumar, 1987] D. Reisis and V. K. Prasanna-Kumar. Parallel processing of image and stereo matching using linear segments. Technical Report IRIS 204, University of Southern California, February 1987.
- [Rice and Jamieson, 1985] T. A. Rice and H. Jamieson. Parallel processing for computer vision. In S. Levialdi, editor, *Integrated Technology for Parallel Image Processing*, pages 57-78. Academic Press, 1985.
- [Rosenfeld and Kak, 1976] A. Rosenfeld and A. Kak. *Digital Image Processing*. Academic Press, New York, 1976.
- [Rosenfeld and Smith, 1981] A. Rosenfeld and R. C. Smith. Thresholding using relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(5):598-606, September 1981.
- [Rosenfeld et al., 1976] A. Rosenfeld, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6(6):420-433, June 1976.
- [Rosenfeld et al., 1986] A. Rosenfeld, B. Simpson, and Squires S., editors. *Proceedings of the DARPA Workshop on Architectures for Image Understanding*, McLean, Virginia, November 1986.
- [Rutkowski et al., 1981] W. S. Rutkowski, S. Peleg, and A. Rosenfeld. Shape segmentation using relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(4):368-375, July 1981.
- [Stout, 1988] Q. F. Stout. Mapping vision algorithms to parallel architectures. *Proceedings of the IEEE*, 76(8):982-995, August 1988.
- [Sunwoo and Aggarwal, 1989] M. H. Sunwoo and J. K. Aggarwal. ViSTA: An integrated vision tri-architecture system. Technical Report CVRC TR-89-6-57, University of Texas at Austin, September 1989. Laboratory for Image and Signal Analysis.
- [Terzopoulos, 1986] D. Terzopoulos. Image analysis using multigrid relaxation methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(2):129-139, March 1986.
- [Waltz, 1972] D. Waltz. *Generating Semantic Descriptions from Drawings of Scenes with Shadows*. PhD thesis, Massachusetts Institute of Technology, November 1972. Artificial Intelligence Laboratory.
- [Weems and Levitan, 1987] C. C. Weems and S. P. Levitan. The Image Understanding Architecture. In *Proceedings of the DARPA Image Understanding Workshop*, pages 483-496, February 1987.
- [Weems, 1988] C. C. Weems. Some sample algorithms for the image understanding architecture. In *Proceedings of the DARPA Image Understanding Workshop*, pages 127-128, April 1988.

Selective Attention as Sequential Behavior: Modeling Eye Movements with an Augmented Hidden Markov Model

Raymond D. Rimey and Christopher M. Brown*

The University of Rochester
Computer Science Department
Rochester, New York 14627

Abstract

Visual attention is an important problem in computer vision that has received little direct attention in computer vision research. We are pursuing a long-range program of research into the visual attention problem. This paper describes our first thrust in this program, the sequential aspect of attentive vision systems. We assume a spatially variant sensor with a fovea and periphery, and study foveal sequencing, or the problem of where next to concentrate high-resolution vision. We present a new explicit representation of attentional sequencing, called an augmented hidden Markov model (AHMM). An AHMM can model a sequence of *where* to place the fovea, or a sequence of *what* objects to look at. A dual-AHMM, combining a *what* and a *where* model with symmetric feedback, is also presented. Our model allows behavior to be learned and to be responsive to individual scene variations.

1 Overview

A system using real-world visual input for decision making must ignore the irrelevant, attend to the salient, and place reliable priorities on tasks and resources. Complexity analysis of the problem of matching visual images to models reveals that *pure parallelism is not enough* to overcome the visual computational burden, but that structure, in the form of a hierarchy of spatial resolutions and of abstractions, can render this visual task tractable [Tsotsos, 1987].

One way human systems overcome the visual computational burden is by the mechanism of *attention*, a topic extensively studied in the areas of psychology and neuroscience. We are pursuing a long-range program of research into the selective attention problem in computer vision.

We are currently exploring the thesis that attention provides mechanisms that control the *allocation* of visual processing preferentially within a scene, leaving open

just which resources are being managed. Here we investigate the allocation of a spatially variant sensor such as one with a *peripheral visual field* of low resolution but wide angle, and a *high-resolution fovea* centered in the visual field. We present a novel approach to learning, representing, and generating explicit *attentional sequences* that direct such a sensor to view specific areas of a scene. The capability we describe is like a visual-motor skill — it emphasizes the efficient acquisition and use of relevant behavior for a repetitively-occurring situation, with the capability of adapting both to individual variations between problem instances and to slow variations in the expected situation. In a usual data-driven foveation (or region of interest) system, a sequence of eye movements *emerges* from a program reacting to the image data (often using low-level saliency, or “interest”, operators). Our *explicit* model for such sequences can, for example, be trained on emergent sequences generated by other algorithms and be made to generate these sequences explicitly, whether they are continuous attentional paths or discontinuous, saccadic fixations. In other words the sequential attentional behavior can become automatic, or “compiled” into a lower-level, pre-attentive visual skill.

Our model is based on the hidden Markov model (HMM), which is roughly a generalization of a teachable, probabilistic finite state automaton. The HMM camera control model operates in a mode oblivious to the visual data. We introduce a modified model, called an augmented HMM (AHMM), for the more typical case in which the movements should be responsive to (*i.e.* be modified by) visual cues.

Three models are described below. The first uses external feedback to affect the AHMM outputs (only). The second uses internal feedback to modify the internal parameters (probabilities) of the AHMM, thus affecting the generation likelihoods directly. These models can be applied for the purpose of generating a sequence of *where* to point the camera. The model can also be applied to generate a sequence of *what* to look at. The final model combines both a *what* and a *where* part, and a symmetrical scheme in which the two parts dynamically feed back to and support each other.

This paper is organized as follows. Section 2 discusses why attention, spatially-variant sensors, and attentional sequencing are important topics in computer vision. Section 3 describes in detail the AHMM models we have

*This material is based on work supported by the NSF under grant CDA-84-22724, and by DARPA/AFOSR research contract no. AFOSR-89-0222. The government has certain rights in this material.

developed, and presents experiments applying the models. Conclusions and plans for future work are given in Section 4.

2 Motivation

Recently the advent of sophisticated controllable visual hardware has emphasized sensor control, or the "active vision" paradigm (e.g. [Bajcsy, 1988, Ballard, 1989, Brown, 1988, Burt, 1988]). Besides advocating the idea that vision and action modules should be designed to cooperate and support each other, active vision promotes a general re-examination of certain aspects of the human visual system [Ballard, 1989] for ideas on how to build computer vision systems. This section discusses two topics in this vein, anthropomorphic visual sensors and visual attention. Then the ideas of attentional sequencing and visual skills are presented.

2.1 Foveal and peripheral sensors

The fovea-periphery distinction is quite dramatic in the human visual system, but usually humans are not consciously aware of it. All of our high-resolution vision is performed by a fovea whose field is only 0.5 degrees of visual arc (about the extent of a quarter coin held at arms' length). The remaining large peripheral field only provides low-resolution vision.

A fovea, accompanied with attentional algorithms and control machinery for directing it, becomes a viable engineering solution when imaging, transmission, and computing bandwidth are limited. Several visual computations may also become easier in the context of a foveal sensory system. For example, various uses of stereo disparity become easier when coupled to camera vergence [Coombs, 1989], as do kinetic depth computations when coupled with fixation [Ballard and Ozcandarli, 1988].

A spatially variant sensory device can be created in several ways. Anthropomorphic VLSI sensors are being constructed (e.g. [Tistarelli and Sandini, 1990]). Software and hardware resolution pyramids are a classic technique (e.g. [Burt, 1988]). Another choice is simply to use two cameras with two different focal lengths. Finally, a simple electronic window could be used.

2.2 Attention

The topic of visual attention has been identified and extensively studied by researchers in psychology and the neurosciences ([Poggio and Hurlbert, 1985, Posner and Presti, 1987, Humphreys and Bruce, 1989]). Attention is usually identified with the limited availability of resources. In the spotlight model of attention a fixed size "spotlight" can be shifted around to enhance visual processing in the area it covers. Covert attention is shifted within the visual field, but is not associated with eye movements, whereas shifts in overt attention are directly linked with eye movements. "Popout" phenomena, studied by Treisman (e.g. [Treisman and Gelade, 1980, Treisman, 1985]) and others, examine the relation between preattentive immediate vision and serial attentive vision.

Ullman's "Visual Routines" paper [Ullman, 1984] explores how ideas from the psychological and neuroscience

research on visual attention can be applied to computer vision, and it enunciates several interesting ideas that have inspired our work. Ullman proposes five primitive visual routines: attention shift (i.e. controlling and moving the location of attention), indexing (i.e. selecting specific locations for further processing), bounded activation, boundary tracing, and marking. Attention shift and indexing are the primitive routines that mainly concern us.

2.3 Attention as sequential behavior: Foveal or attentional sequencing

The psychological literature contains much work on eye movements, the most direct evidence for visual attention (though usually for two-dimensional stimuli). Yarbus' book [Yarbus, 1967] documents graphic traces of eye movements as humans examine scenes for relatively long times (three minutes). Some intriguing observations are: Subjects always foveate *only* select areas in the scene, those containing "relevant objects". For a single task, a given subject repeatedly uses a foveation sequence with only minor variations. For different tasks, the general sequence that all subjects use is highly dependent on the task.

Yarbus's work led to the idea that an object is represented for visual recognition as a *scanpath* - a time-ordered sequence of features perceived at each fixation, along with motor commands that link the fixations [Noton and Stark, 1971]. Subsequent work [Stark and Ellis, 1981] presented a simple probabilistic model for fixation sequences. Didday and Arbib [Didday and Arbib, 1975] describe how similar behavior can emerge from parallel operations on foveal and peripheral image data. A modern piece of work with a fovea-periphery distinction, controllers to determine the location of the next fixation, and experiments is that of [Bolle *et al.*, 1989]. Three controllers are reported: A simple scanning process to move the fovea in a task- and data-independent way; one whose candidate is the largest so-far unexplained region; and one that tries to resolve conflicts between the interpretation so far and the model database. Similar objectives were accomplished with resolution pyramid hardware [Burt, 1988], where foveation was implemented as coarse to fine search through the pyramid.

Control of a fovea and blending its output over fixations is an increasingly common topic [Abbott and Ahuja, 1989, Browse and Rodrigues, 1988, Yeshurun and Schwartz, 1989]. The control algorithms are typically just to examine next the area of highest interest (using some bottom-up measure), using memory or a salience-reduction method to avoid re-examining an area, and perhaps an exploratory urge. Clark and Ferrier [Clark and Ferrier, 1988] report experiments using a similar scheme for controlling a binocular robot.

2.4 Visual skills

Two aspects of the previous work stand out. First, previous systems have generally produced *sequences of fovea movements*. Given limited resources, sequential allocation is obviously a reasonable strategy. Secondly, almost all previous work has studied *emergent sequences*. Se-

quence steps are produced as the output of processes operating on the image data. They are not represented, remembered, or available in advance.

The alternative is *explicit sequences*: Sequence representations are maintained at some level, and can be modified, retrieved, and generally appear in computations. Explicit representations of foveation sequences are rarely proposed, and seldom implemented. The main exception is the experiments of [Stark and Ellis, 1981], and perhaps the idea of motor programs (e.g. [Wright, 1990]). Indeed, foveation sequences are not a convincing way to represent objects for recognition.

However, explicit sequences may well be a good way to represent a strategy for certain types of skilled observation of a structured environment. Emergent sequences, computed a step at a time, are analogous to motor behavior, mediated primarily by general perception and by more or less cognitive involvement. Emergent foveation sequences must be rederived each time, but should be the same for the same situations. Remembering such a sequence efficiently captures the effects of the cognition applied to the task domain. A remembered sequence can be regenerated while also being fine-tuned using current visual feedback data. Thus, a remembered (explicit) sequence is analogous to skilled motor behavior.

For example, when entering into your new office for the very first time, you look around in an undirected way. You tend to look at "interesting" objects, but also at areas and objects that in retrospect are not of any significance or never change. Over time, you develop a pattern of where you look upon entering your office. Perhaps you first look towards the terminal in the near corner, expecting to see an office mate there, and then across to your desk in another corner, scanning another office mate's desk along the way. (The emergent behavior developed a pattern that has now been learned as an explicit behavior, or habit.) While executing the habitual eye movement behavior you often modify it according to visual cues along the way. For example, you can tell that no one is at the terminal using your peripheral vision, and if not, your eyes begin moving towards your desk sooner. If there is some interesting new object on the office mate's desk or your own, your eyes are attracted to it before continuing along their typical path. (The habitual behavior is modified by cues in visual feedback.)

Other examples abound, for example, in industrial visual inspection or driving skills.

In summary, the skill or habit analogy is appealing. A method to model a visual skill, so it could be learned and executed with visual feedback, would be useful. Since knowledge is "compiled" into a visual skill, simpler vision processing would be sufficient, thus reducing computational demands. The model could also specialize itself and adapt to its environment, compensating for single-instance and slowly trending variations in the world.

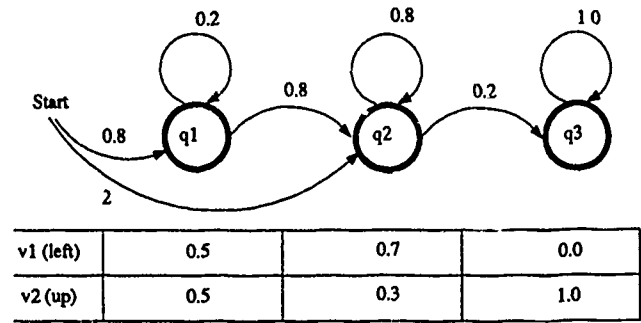


Figure 1: Example of a simple HMM.

3 An augmented hidden Markov model for explicit sequences with visual feedback

This section presents an explicit representation — an augmented version of a hidden Markov model — for attentional sequences modified by visual feedback.

3.1 Hidden Markov models

We need the ability to learn, represent, generate and even classify sequences of (2-D or 3-D) spatial locations for foveations or attention. The hidden Markov model (HMM) has been widely used to classify signals in speech recognition systems, but in fact it has all the abilities we desire. Although it is a very general model for sequences it has not been used much in other fields, such as computer vision. The key points about HMMs are summarized here. For details see the excellent tutorial by Rabiner [Rabiner, 1989, Rabiner and Juang, 1986]. Methods to incorporate feedback into an HMM, the final ability we need, are presented in following sections.

An HMM is somewhat like a probabilistic finite state machine. It is formally defined as $\lambda = (A, \pi, B)$ with states $Q = \{q_1, \dots, q_N\}$ and symbols $V = \{v_1, \dots, v_M\}$. The probability of transitioning at time step t from state q_i to state q_j is given by $A = \{a_{ij}\}$ where $a_{ij} = P(q_j \text{ at } t+1 | q_i \text{ at } t)$. The initial state is determined by $\pi = \{\pi_i\}$ where $\pi_i = P(q_i \text{ at } t=1)$. If the HMM is in state q_j it produces symbol v_k according to the probability $B = \{b_j(k)\}$ where $b_j(k) = P(v_k \text{ at } t | q_j \text{ at } t)$. Note that the symbol sequence is observable, but that the state sequence is "hidden" (not observable).

Figure 1 shows an example HMM. The state sequence it models will tend to contain a short subsequence of q_1 states, a larger subsequence of q_2 states, and q_3 states until the end of the sequence. Assuming that symbols v_1 and v_2 were associated with "left" and "up"-ward (incremental) eye movements, and that the HMM remained in state q_1 for several steps, then it would tend to move the eyes to the upper left during that time.

The graph structure of an HMM may be arbitrary. Graphs with a left-to-right flow, such as in Figures 1, 5 and 6, are often appropriate for non-cyclic sequences. In practice the particular graph structure is crucial to performance and its selection requires experimentation.

HMMs have three associated capabilities, at least one of which is used in any application. They can classify sequences, they can be learned from examples, and they can generate sequences. Each of these capabilities is well known: detailed algorithms for implementing them may be found in [Rabiner, 1989, Rabiner and Juang, 1986] and will not be repeated here.

A single HMM λ_i is associated with each possible class ω_i . An observed sequence O is classified as the most likely class, according to $P(O | \lambda_i)$. The parameters λ_i for each HMM are estimated separately using a "training set" of examples and an algorithm to maximize $P(O | \lambda_i)$ over the set.

Sequences are generated using random numbers chosen from the appropriate probability distributions in λ_i . The length of a sequence is determined by a bound on $P(O | \lambda_i)$ or by using a fixed length. The random method, when *visual feedback* is added, gives robustness to local and long-term variations in the world.

3.2 Using HMMs for oblivious eye movements

To apply HMMs to attentional sequences, the symbols must be related to spatial locations or movements. Following are a few choices.

- *Incremental-position movement.* The symbols are $v_i \in [0, 7]$, the eight chain code ("compass point") directions. The eye rotates a fixed increment so that the image shifts by an increment in the given direction. Here, the attentional sequence is a relatively smooth path, of the sort arising in contour following, doing vision through a reduction tube, or in some other situations [Ullman, 1984].
- *Large-position movement.* The symbols are $v_i = (\theta, R)$ where θ and R are quantized direction and distance. Each movement is relative to the current eye location. Alternatively, the symbols can be $v_i = (x, y)$ where each movement is in a fixed coordinate system. Here, an eye movement sequence is similar to a series of saccades.
- *Object sequencing.* The symbols are feature vectors describing objects. If image analysis produces such feature vectors throughout the image, the sequence defines a series of locations (possibly ambiguous, see Section 3.5) in the image.

In all cases a training set of example sequences can be created from the emergent movement decisions output by some other algorithm, or a human using a pointer on several example images. Alternatively, if technical considerations permit, recordings of human eye fixations generated during the task can be used.

3.3 Augmented hidden Markov models: external feedback modification of oblivious eye movements

This section presents a trivially augmented AHMM, which will serve as an example to explain feedback modified eye movements and saliency-based feedback cues. The following sections present more sophisticated AHMMs.

The external feedback AHMM. Let the oblivious sequence of symbols normally generated by an HMM be $O = O_1, \dots, O_T$, and let the feedback sequence be $S = S_1, \dots, S_T$. S_i is a symbol representing the feedback cue at index i in the sequence. We assume that feedback symbols and output symbols are both of the same type. The AHMM simply outputs the feedback modified path, $M = M_1, \dots, M_T$, according to the following equation.

$$M_i = \begin{cases} S_i & \text{with probability } \alpha \\ O_i & \text{otherwise} \end{cases} \quad (1)$$

α is a parameter from 0.0 to 1.0 that regulates the amount of influence from the feedback sequence. With $\alpha = 0.0$ the feedback data is completely ignored. With $\alpha = 1.0$ the HMM is ignored and the feedback sequence is tracked exactly.

Computation of visual cues. In our experiments we have used feedback cues based on local maxima of a saliency image. S_i is simply the direction (actually a chain code symbol) towards the local maximum of the saliency data. Generally, of course, any other kind of feedback cue can be used. A saliency image is computed as a weighted sum of several feature images, which are themselves computed from the original intensity image.

A Gaussian neighborhood function can be used to compute the local maximum. However, this would cause problems when generating incremental-position ("smooth path") eye movements, since there would be a tendency for the path to turn around towards a maximum behind it and not depart from a maximum it had reached. We handle the problem by using a neighborhood function that emphasizes saliency points in the direction of the path, and saliency points away from the current location.

The above method is applied separately to the foveal and peripheral images to compute a peripheral feedback cue $S_i^{(p)}$ and a foveal cue $S_i^{(f)}$.

Modifying incremental-position movement. Recall that in an incremental-position movement application each O_i is a chain code symbol. Before the eye moves, this sequence defines a path through the peripheral image. First, a periphery modified path, with elements $M_i^{(p)}$, is generated according to

$$M_i^{(p)} = \begin{cases} S_i^{(p)} & \text{with probability } \alpha \\ O_i & \text{otherwise} \end{cases} \quad (2)$$

Then, foveal modification is performed according to

$$M_i^{(f)} = \begin{cases} S_i^{(f)} & \text{with probability } \beta \\ M_i^{(p)} & \text{otherwise} \end{cases} \quad (3)$$

resulting in the final sequence $M^{(f)} = M_1^{(f)}, \dots, M_T^{(f)}$. Note that the eyes must actually be moved to make the appropriate foveal data available to compute $S_i^{(f)}$.

Modifying large-position movement. Large-position movement might, for example, use symbols of the form $O_i = (x, y)$, which denote absolute positions ("targets") in the periphery. Feedback is determined from the maximum of the peripheral saliency image in

the neighborhood of the target. The maximum is computed using a Gaussian neighborhood function centered on the target. Unfortunately equation (1) does not provide a satisfactory form of modification. The AHMM presented in the next section addresses this problem. (Modifications using foveal data before a large-position movement is performed are not appropriate since the target is usually outside the fovea. However, once the target is reached, the foveal data can be used to perform a small adjustment movement.)

3.4 An augmented hidden Markov model with internal feedback

The internal feedback AHMM has parameters that vary as a function of time, denoted as $\lambda^{t+1} = (A^{t+1}, \pi, B^{t+1})$. The AHMM operates as follows. Assume that the AHMM is at time step t , that it has already output a sequence of symbols O_1, \dots, O_t , and that the current state is q_i . The feedback symbol S_t is available, where the value of S_t is v_k . S_t is used to modify λ^t into λ^{t+1} , then the AHMM uses λ^{t+1} to generate the next symbol, O_{t+1} . Obviously the choice of the next generated symbol depends partially on what the few most recent feedback symbols have been. The λ^t parameters also slowly decay over time to their original values. So as long as consistent feedback symbols are available, their effect on the AHMM parameters will endure, but eventually the parameters return to their original values. The initial state probabilities π do not vary in time since a feedback symbol is not available at time $t = 0$. The equations for computing λ^{t+1} are summarized below. See [Rimey and Brown, 1990] for their derivation and a more complete explanation.

A weighting factor. The equations for modifying the AHMM parameters use three key values: i , k , and w_j^t . The value i is determined from the state q_i of the AHMM at the current time step, t . The value k is determined from the value of the current feedback symbol, $S_t = v_k$. The values for i and k are known and will be assumed in all the equations below.

The final key value is w_j^t , a probabilistic weight computed from i and k : w_j^t is the probability of being in state q_j at time $t + 1$, given the information that the AHMM is in state q_i at time t and will output symbol v_k at time $t + 1$. The equation for computing w_j^t is

$$w_j^t = \frac{a_{ij}^t b_{ij}^t(k)}{\sum_{i=1}^N a_{ii}^t b_{ii}^t(k)} \quad 1 \leq j \leq N. \quad (4)$$

The current feedback symbol (S_t) is assumed to be a prediction of the next output symbol (O_{t+1}), so w_j^t provides an indication of how consistent the immediately possible state transitions are with the current feedback, and it can be used to bias the state transition probabilities.

Modification of a_{ij}^t . New values for the transition probabilities, denoted by \tilde{a}_{ij}^{t+1} for the time being, that are most consistent with the feedback are $\tilde{a}_{ij}^{t+1} = w_j^t$. Generally it seems desirable to modify a_{ij}^t slowly rather than completely replace it. Therefore only a fraction, $r_f w_j^t$, is mixed with the current value. The new equation is

$$\tilde{a}_{ij}^{t+1} = r_f w_j^t + (1 - r_f) a_{ij}^t \quad 1 \leq j \leq N \quad (5)$$

where r_f ($0 \leq r_f \leq 1$) is a modification gain. Larger gain values emphasize the feedback modified transition probability over the original probability.

Modification of $b_j^t(l)$. The emission probabilities in the AHMM must be updated for each state q_j that can be reached in one time step from the current state q_i . The equations for the updated emission probabilities, denoted $\tilde{b}_j^{t+1}(l)$, and already incorporating a mixing gain, are as follows.

$$\tilde{b}_j^{t+1}(l) = \frac{e_j^t(l)}{\sum_{m=1}^M e_j^t(m)} \quad 1 \leq j \leq N, \quad 1 \leq l \leq M \quad (6)$$

$$e_j^t(l) = s_f w_j^t d_j^t(l) + (1 - s_f) b_j^t(l) \quad (7)$$

$$d_j^t(l) = \begin{cases} 1.0 & \text{if } l = k \\ 0.0 & \text{otherwise.} \end{cases} \quad (8)$$

The denominator in equation (6) ensures that $\tilde{b}_j^{t+1}(l)$ is a valid probability. The modification gain is s_f ($0 \leq s_f \leq 1$), where small gain values emphasize the original emission probability over the feedback modified ones. The key term in these equations, the one contributed by the feedback, is $w_j^t d_j^t(l)$.

Time decay. Equations (5) and (6) are the basis for modification at any instant in time. Since these modifications should only be maintained as long as they are justified by feedback information, the modifications are made to decay in time as follows

$$a_{ij}^{t+1} = r_d \tilde{a}_{ij}^{t+1} + (1 - r_d) \hat{a}_{ij} \quad 1 \leq j \leq N \quad (9)$$

$$\tilde{b}_j^{t+1}(l) = s_d \tilde{b}_j^{t+1}(l) + (1 - s_d) \hat{b}_j(l) \quad 1 \leq j \leq N, \quad 1 \leq l \leq M \quad (10)$$

where \hat{a}_{ij} and $\hat{b}_j(l)$ are the original values, which do not change over time. The decay gains are r_d and s_d ($0 \leq r_d, s_d \leq 1$). Small decay gains cause the probabilities to decay quickly to their original values. These equations give the final values for a_{ij}^{t+1} and $b_j^{t+1}(l)$.

Multiple feedback signals. The above AHMM can easily be extended to the case that several different feedback symbols are available from different feedback sources. The feedback symbols at time t are denoted by a set S_t . For example, this set could contain either simultaneous peripheral and foveal feedback symbols, or multiple peripheral feedback symbols. The updating equations are similar to those above, and can be found in [Rimey and Brown, 1990].

Modifying incremental-position movement. The above AHMM can be used in a straight-forward manner to generate incremental-position movements. Multiple feedback symbols, for simultaneous peripheral and foveal feedback, are used.

Modifying large-position movement. The above AHMM can be more easily applied to large-position movements than could the first AHMM we presented. The AHMM operates in two stages during each time step. First it generates a preliminary output symbol (target location). Then a Gaussian neighborhood function is centered on the target location. Local maxima are detected and used as multiple feedback to modify

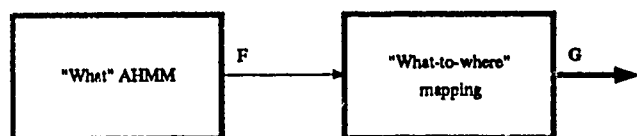


Figure 2: The "what" version of the AHMM. A Darker signal path denotes a set rather than a single signal value.

the AHMM parameters. Lastly, the AHMM generates the final version of the output symbol for the time step. The internal feedback modifications in the AHMM result in a form of (application independent) averaging of the feedback symbols and the "oblivious" output symbol, essentially because the internal probabilities are averaged over time. Multiple feedback paths can also be used, for example, by using several of the largest local peripheral-saliency maxima (rather than just one). Large-position movements using foveal feedback are still not possible because the fovea does not view the target area.

Adding new states. The above AHMM does not permit an existing state to add a new non-zero emission probability, or to add a new link, or for the graph to add a new state. The modification equations above were derived assuming that the feedback sequence reflects variations that are generally still consistent with the underlying modeled sequence. If the underlying sequence (in the real world) has changed in a fundamental but local way it may be necessary to insert a new state into the AHMM graph. This situation may be detected by a small value of

$$\max_{1 \leq j \leq N} \hat{a}_{ij}^t \hat{b}_j^t(k) \quad (11)$$

in which case a slow semi-permanent modification may be initiated in which a new state is added.

3.5 A mutually supporting what-where feedback model constructed from two internal feedback AHMMs

This section describes a hybrid AHMM that elegantly incorporates both "what" and "where" sequence models and uses them to play off and mutually support each other.

A what-AHMM. The "what" version of the AHMM, called a what-AHMM and shown in Figure 2, contains two sections. The first section is an internal feedback AHMM whose output symbols F_t , called *what-symbols*, are feature vectors intended to describe an object or characteristics of objects. Such feature vectors are assumed to have been computed for each pixel in the peripheral image. The second section of the what AHMM performs a "what-to-where" mapping, meaning that it maps a feature vector into the set of image coordinates G_t in the current image where instances of those feature vectors (or similar ones) exist. Actually it maps to the eye movement commands, called *where-symbols*, that would cause those locations in the image to be centered on the fovea.

If each G_t contains exactly one element, the output sequence will fixate the desired objects in the scene. Each

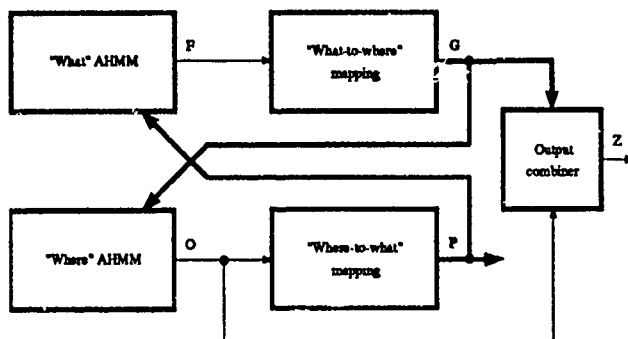


Figure 3: The "what-where" version of the AHMM. A Darker signal path denotes a set rather than a single signal value.

G_t does not generally contain exactly one element, so some method must be developed to select among the choices. One option is to use a where-AHMM to help pick among the choices. In fact, the what-AHMM can be made to help the where-AHMM with its *own* choices.

A what-where-AHMM. The what-where-AHMM shown in Figure 3 contains three distinct parts: a what-part and a where-part which both feed back to each other, and an output combiner. The what-part is like the what-AHMM described above. It uses feedback, but the feedback is a sequence of *sets* of what-symbols P_t , the output of the where-part at time step t .

The where-part contains two sections, similar to those in the what-part. First it has an internal feedback AHMM, which outputs a sequence of where-symbols O_t . In this model the where-symbols are intended to be large-position movement symbols. Secondly it has a "where-to-what" mapping which determines for each where-symbol O_t the location in the current image it corresponds to, and outputs the set of feature vectors P_t in that local area of the image. The AHMM in the where-part uses as feedback a sequence of sets of where-symbols G_t , which is the output of the what-part.

Finally, the output combiner determines the overall output of the what-where-AHMM. The overall output at time step t is Z_t , a where-symbol (i.e. an eye movement command), selected as the element of the set G_t that has the smallest distance to the symbol O_t .

Operation of the what-where-AHMM is as follows. At each time step, each of the two parts produces a set of feedback symbols that reflects its own preference for action. Each then updates its own preferences taking the other's into account, and then generates its own final preference for action at that time step. The set of final preferences is reduced to a single output symbol by the output combiner.

Incorporating high-resolution (foveal) feature vectors. So far, the what-where-AHMM has used only peripheral image data so its feature vectors (what-symbols) should be considered to be low-resolution feature vectors. After each eye movement, new fovea data is available to compute a high-resolution feature vector, essentially a verification of what the low-resolution feature vector suggested might be at that location. A negative

verification might be used to modify further the AHMM, for example, to move to a new location containing one of the other instances of the same feature vector.

3.6 Experiments

The AHMM eye movement models have been implemented using the Rochester Head and its associated image processing hardware [Brown, 1988]. Computer control of the two cameras on the Head permits individual camera pans, a shared tilt, and either smooth path or saccadic movements. See [Rimey and Brown, 1990] for a complete description of our experiments.

Figure 4 shows a Lab scene typical of those used in the experiments — a table top with a variety of objects. All figures here contain both peripheral and foveal components: The majority of a figure is a low resolution peripheral image (128x128, zoomed 4x), while the center contains the high resolution foveal image (also 128x128). For visual cues in these experiments we used a simple saliency image that was easy to implement, the equally weighted sum of five features derived from the Sobel edge operator and the grayscale variance. The graphics superimposed on all figures illustrate the points which would be fixated if the cameras were to execute a movement sequence. Generation of an oblivious or peripheral modified sequence does not require camera movement, whereas a foveal modified sequence does require it.

External feedback AHMM, incremental-position movement. Figure 5 shows the graph structure of the AHMM used. A mouse was used to create a training set containing 30 sequences. An AHMM was trained on that set and used to generate the oblivious path shown in white in Figure 4. Such a path might correspond to knowing the desired object is normally kept on the left side of the desk. Peripheral image data modified the oblivious path, resulting in the path shown in gray. Here the peripheral data keeps the path from overshooting the stuffed animals. However, later it does not effectively pull the path closer to either the soda cans or the pile of small parts, as would be preferred. Finally, the AHMM was run also using foveal data modification, obtaining the path shown in black. The foveal saliency data attracts the latter half of the path to the small pile of parts. The two modified paths in these results were produced with the peripheral gain $\alpha = 0.3$ and the foveal gain $\beta = 0.4$ (equations (2)-(3)). Experiments with larger and smaller values for α and β have verified that the model can provide more and less aggressive path modification.

External feedback AHMM, large-position movement. The large-position movement experiments use AHMM symbols that are (coarsely quantized) absolute retina positions, (x, y) . Figure 6 shows the AHMM graph structure used. An algorithm similar to that in [Clark and Ferrier, 1988] was used to generate 20 training examples, thus showing how the AHMM can learn the emergent behavior generated by some other algorithm. In this case, the other algorithm iteratively fixated the point in the image with a maximum saliency value, zeroed out the local saliency around that maximum, and then went to the next largest maximum, until 5 fixa-

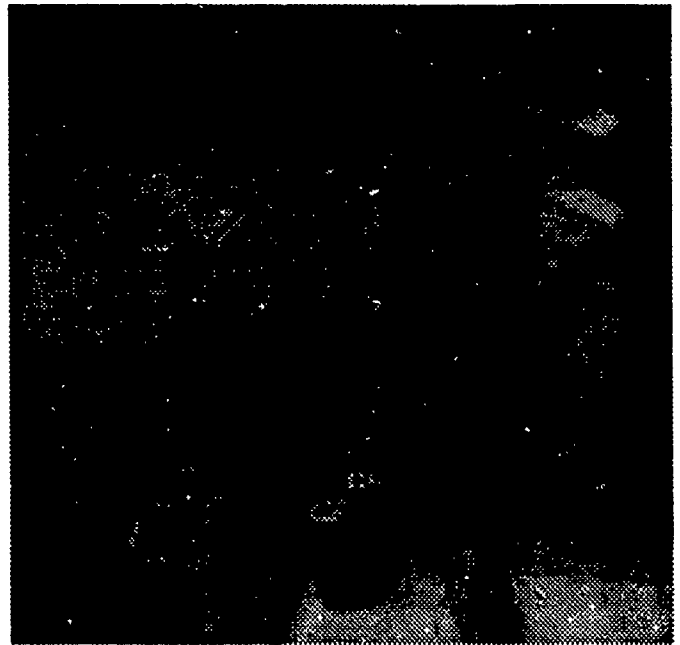


Figure 4: Incremental-position movement sequence. Parameters: $\alpha = 0.3$ and $\beta = 0.4$. Oblivious path (white), peripheral (gray) and foveal (black) modified paths.



Figure 5: AHMM graph for incremental-position movement experiment.

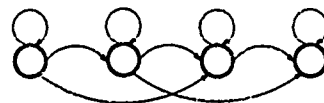


Figure 6: AHMM graph for large-position movement experiment.

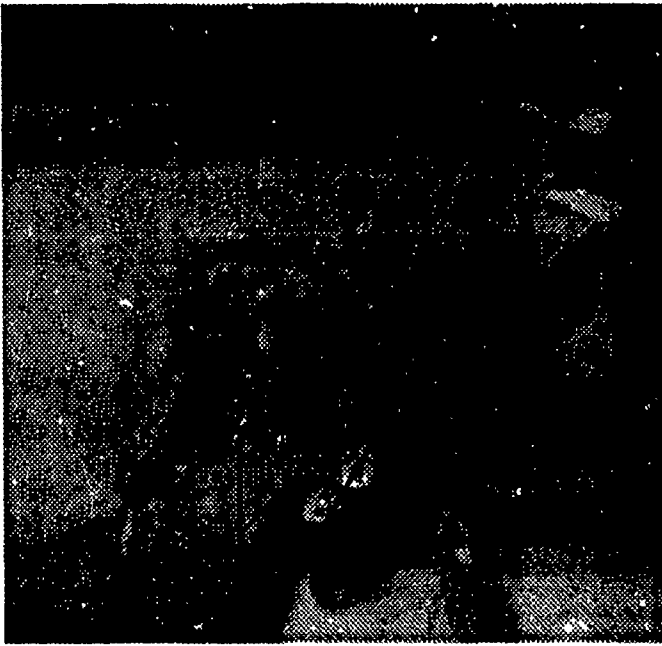


Figure 7: Large-position movement sequence generated using AHMM which was trained on examples produced by "other" algorithm. Oblivious path (white) and periphery modified path (gray).

tions were made. Figure 7 shows an oblivious sequence (white) generated by the trained AHMM, and the sequence modified using peripheral salience (gray). Note how the modified sequence has been drawn to the locally more interesting areas of the image.

Internal feedback AHMM, incremental-position movement. The internal feedback AHMM has been investigated with experiments parallel to the incremental-position experiments for the external feedback AHMM.¹ The same AHMM graph structure (Figure 5) was used. The same trained AHMM parameters were also used, except that any zero valued probabilities were changed to have very small non-zero values. These AHMM parameters served as the initial (fixed in time) version of the model. The saliency image used in these experiments was simply the Sobel edge magnitude image.

Figures 8-10 show the result of introducing varying amounts of foveal feedback into the model.² In these experiments the gain value of $r_{f,(f)}$ was set equal to $s_{f,(f)}$ and varied over the values 0.2, 0.5 and 0.8. The 0.2 value results in a path almost identical to the original, while the 0.5 value results in a path that begins to be drawn more towards the nearby objects in the scene (the soda cans) - a reasonable behavior, although perhaps not the best one. A value of 0.8 results in even stronger modifi-



Figure 8: Incremental-position movement sequence, small foveal feedback gain. Gains are: $r_{f,(p)} = s_{f,(p)} = 0$, $r_{f,(f)} = s_{f,(f)} = 0.2$, $r_d = s_d = 0.9$.

cations, producing a path that is immediately drawn to the detergent boxes, however the effect of the AHMM's trained behavior eventually gains control and the path resumes its course to the lower left towards the stuffed animals.

4 Conclusions

Summary. We are pursuing a long-range research program with the goal of isolating, clearly defining, and studying selective attention as a problem area in its own right. In this paper we assume a spatially variant sensor, such as a fovea and periphery, and study some aspects of attentional sequences. An augmented hidden Markov model (AHMM) is presented as a way to model explicit eye movement sequences while incorporating feedback from visual cues. AHMMs can deal with sequences of locations (where) or of object characteristics (what) or even both (dual what/where). A more detailed presentation of the theoretical and experimental results reported here can be found in [Rimey and Brown, 1990].

We conclude that AHMMs indeed are a promising way to acquire, represent, generate, and use probabilistic sequences for computer recognition. The AHMM is not a replacement for high- or low- level approaches to computing where or what to look at next. It provides a mechanism like a visual skill, for remembering how to allocate the visual sensor, but it is only one of several mechanisms (each with limited uses) that an attentive vision system might contain.

HMMs in computer vision. The HMM is a fairly general yet quite simple model and as such deserves consideration and investigation in areas other than speech

¹These experiments are still in progress.

²These sequences are different than Figure 4 only because a different random number sequence and a slightly different scene was used when each set of these experiments was performed. The same random number sequence was used for each of the experiments shown in Figures 8-10.



Figure 9: Incremental-position movement sequence, medium foveal feedback gain. Gains are: $r_{f,(p)} = s_{f,(p)} = 0$, $r_{f,(f)} = s_{f,(f)} = 0.5$, $r_d = s_d = 0.9$.

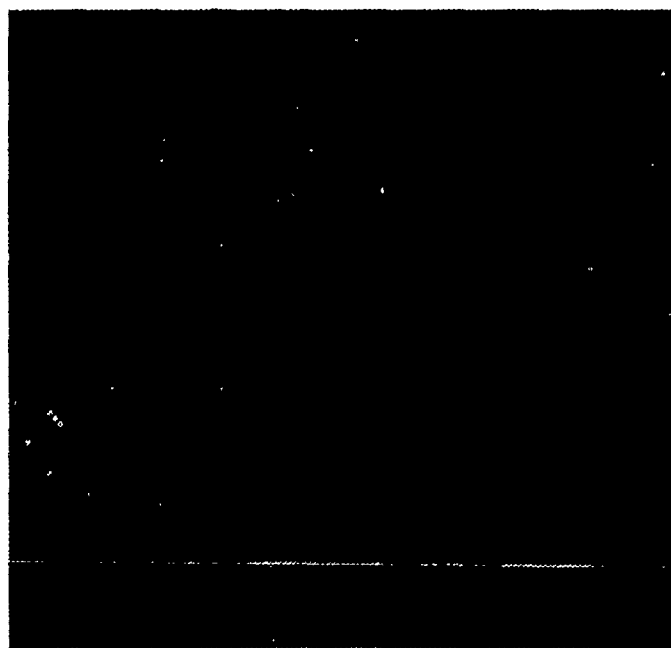


Figure 10: Incremental-position movement sequence, large foveal feedback gain. Gains are: $r_{f,(p)} = s_{f,(p)} = 0$, $r_{f,(f)} = s_{f,(f)} = 0.8$, $r_d = s_d = 0.9$.

understanding. To help illustrate the HMM's usefulness this section briefly mentions a few other problems to which it can be applied.

Classification of certain time-varying patterns is another application, *e.g.* recognizing non-rigid objects through their motion or temporal-texture characteristics. Another capability is object classification from view sequences. Here a view is characterized by a feature vector, the viewpoint can be rotated around the object, and the HMM can essentially (and compactly) learn the *topology* of features over a subset of the sphere of viewpoints (*i.e.* the object's feature-aspect graph). Finally, the HMM can be viewed as a trainable finite state machine, which may be a useful characterization for those AI researchers who are hand-crafting finite state machine variants.

Directions for future work. Our work on the AHMM has helped us to concentrate on the concept of attentional sequences. Our intention is to pursue similar efforts to learn more about other concepts important to an attentive computer vision system. Some related concepts we are considering are: visual masking, perceptual grouping, figure-ground separation, and the role of structure in vision. We are also interested in studying: Bayesian, decision theoretic frameworks, incremental and deictic representations, and real-time scheduling of tasks that can only compute partial results. Our current plan is (1) to continue developing and experimenting with the what-where-AHMM, and (2) to start a new thrust, studying attention as allocation and scheduling.

Acknowledgements

We thank the entire Rochester vision group for comments on a early presentation of this work. Dana Ballard provided the key inspiration that led to the what-where-AHMM. He, as well as David Coombs, Steve Whitehead, and Mike Swain provided valuable comments on the written report.

References

- [Abbott and Ahuja, 1989] A. L. Abbott and N. Ahuja. Surface reconstruction by dynamic integration of focus, camera vergence, and stereo. In *Proceedings: International Conference on Computer Vision*, pages 532-543, 1989.
- [Bajcsy, 1988] R. Bajcsy. Active perception. *IEEE Proceedings*, 76(8):996-1005, August 1988.
- [Ballard and Ozcanlarli, 1988] Dana H. Ballard and Altan Ozcanlarli. Eye movements and visual cognition: Kinetic depth. In *Proceedings. International Conference on Computer Vision*, December 1988.
- [Ballard, 1989] D. H. Ballard. Reference frames for animate vision. In *Proceedings: International Joint Conference on Artificial Intelligence*, pages 1635-1641, 1989.
- [Bolle *et al.*, 1989] R. M. Bolle, A. Califano, and R. Kjeldsen. Data and model driven foveation. In *IBM Research Report*, October 1989.

- [Brown, 1988] C. M. Brown. The Rochester robot. Technical Report 257, Department of Computer Science, University of Rochester, August 1988.
- [Browse and Rodrigues, 1988] R. A. Browse and M. G. Rodrigues. Propagation of interpretations based on graded resolution input. In *Proceedings: International Conference on Computer Vision*, pages 405-410, 1988.
- [Burt, 1988] P. J. Burt. Smart sensing within a pyramid vision machine. *IEEE Proceedings*, 76(8):1006-1015, August 1988.
- [Clark and Ferrier, 1988] J. J. Clark and N. J. Ferrier. Modal control of an attentive vision system. In *Proceedings: International Conference on Computer Vision*, pages 514-523, 1988.
- [Coombs, 1989] David J. Coombs. Tracking objects with eye movements. In *Proceedings: OSA Topical Meeting on Image Understanding and Machine Vision*, 1989.
- [Didday and Arbib, 1975] R. L. Didday and M. A. Arbib. Eye movements and visual perception: a two visual system model. *International Journal Man-Machine Studies*, 7:547-569, 1975.
- [Humphreys and Bruce, 1989] G. W. Humphreys and V. Bruce. *Visual Cognition: Computational, Experimental, and Neuropsychological Perspectives*. Lawrence Erlbaum, 1989.
- [Noton and Stark, 1971] D. Noton and L. Stark. Eye movements and visual perception. *Scientific American*, 224(6):34-43, 1971.
- [Poggio and Hurlbert, 1985] T. Poggio and A. Hurlbert. Spotlight on attention. *Trends in Neuroscience*, pages 309-311, July 1985.
- [Posner and Presti, 1987] M. I. Posner and D. E. Presti. Selective attention and cognitive control. *Trends in Neuroscience*, 10:13-17, 1987.
- [Rabiner and Juang, 1986] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, January 1986.
- [Rabiner, 1989] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *IEEE Proceedings*, 77(2):257-286, February 1989.
- [Rimey and Brown, 1990] R. D. Rimey and C. M. Brown. Selective attention as sequential behavior: Modeling eye movements with an augmented hidden Markov model. Technical Report 327 (revised), Department of Computer Science, University of Rochester, April 1990.
- [Stark and Ellis, 1981] L. Stark and S. R. Ellis. Scanpaths revisited: Cognitive models direct active looking. In D. F. Fisher, R. A. Monty, and J. W. Senders, editors, *Eye Movements: Cognition and Visual Perception*. Lawrence Erlbaum, 1981.
- [Tistarelli and Sandini, 1990] M. Tistarelli and G. Sandini. Robot navigation using an anthropomorphic sensor. In *Proceedings: IEEE International Conference on Robotics and Automation*, 1990.
- [Treisman and Gelade, 1980] A. M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12:97-136, 1980.
- [Treisman, 1985] A. Treisman. Preattentive processing in vision. *Computer Vision, Graphics, and Image Processing*, 31:156-177, 1985.
- [Tsotsos, 1987] J. K. Tsotsos. A 'complexity level' analysis of immediate vision. *International Journal of Computer Vision*, 1(4):303-320, 1987.
- [Ullman, 1984] S. Ullman. Visual routines. *Cognition*, 18:97-157, 1984.
- [Wright, 1990] C. E. Wright. Controlling sequential motor activity. In D. N. Osherson, S. M. Kosslyn, and J. M. Hollerbach, editors, *An Invitation to Cognitive Science, Volume 2, Visual Cognition and Action*, pages 285-316. MIT Press, 1990.
- [Yarbus, 1967] A. Yarbus. *Eye movements and vision*. Plenum Press, New York, 1967.
- [Yeshurun and Schwartz, 1989] Y. Yeshurun and E. L. Schwartz. Shape description with a space-variant sensor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1217-1222, November 1989.

Satisfying the Resolution Constraint in the "MVP" Machine Vision Planning System

Konstantinos Tarabanis*
Computer Science Department
Columbia University
New York, New York 10027

Roger Y. Tsai
Manufacturing Research
IBM T.J. Watson Research Center
Yorktown Heights, New York 10598

Peter K. Allen
Computer Science Department
Columbia University
New York, New York 10027

Abstract

In this paper we present techniques to analytically determine the complete locus of camera poses and optical settings that satisfy the resolution requirements of a machine vision task for given features of interest. This work is part of more extensive research that we are pursuing, as part of our "MVP" (Machine Vision Planning) system, on the problem of sensor planning for satisfaction of several generic machine vision requirements, one of which is resolution. It is therefore important to determine the entire admissible domain of sensor locations and settings for each task constraint, so that these component results can be combined in order to find globally admissible sensor parameter values. Resolution is considered a key factor for machine vision tasks as it determines the accuracy of the reported measurements. Therefore, when designing a vision system that will satisfy the resolution requirements of the machine vision task at hand, it is necessary to properly select the image sensor (e.g. pixel size), as well as decide its placement and

settings. We present techniques to determine the latter two, namely, admissible camera poses and settings, using a synthesis approach to the problem. This approach improves on the generate-and-test techniques currently employed in which sensor configurations are generated and then tested for satisfaction of the task criteria. In this work, all five degrees of freedom of camera placement are considered and thus the results are applicable to a general three-dimensional viewing configuration. Camera placement experiments are shown that demonstrate the method in an actual robotic setup. A camera mounted on a robotic arm is placed and focused according to the results of the new technique and camera views are taken to verify that the feature of interest is visible, within the camera field of view and resolvable to the given specification. Results of this research will help automate the vision system design process, assist in programming the vision system itself and lead to intelligent automated robot imaging systems.

1 PROBLEM

Automation in manufacturing is presently dominated by special-purpose machines that perform predetermined functions in prespecified and tightly controlled environments. Since these systems are clearly inflexible and generally cost a great deal, considerable interest has been drawn to flexible sensor-based automation systems that are able to carry out a variety of functions on the manufacturing floor in a more flexible working environment and at lower cost. Such systems are equipped with various types of sensors in order to interact in an intelligent and flexible manner with the environment. However, despite this added intelligence many functions still require substantial human involvement. For instance, determining the appropriate *configuration of the sensors* as well as *programming the sensor-based system*

itself remain as very human-intensive operations that considerably increase the development time, cost and complexity of such flexible manufacturing systems. It is functions such as these that this research attempts to automate, generating as a result more flexible and autonomous sensor-based systems.

In particular, we are investigating the problem of *model-based and task-driven sensor planning* (Figure 1). That is, by making use of models of the sensors and the objects in the environment, sensor parameters are automatically determined that satisfy generic requirements of sensor-based tasks. More specifically, this problem involves developing planning algorithms that will automatically generate admissible parameter values, such as sensor locations

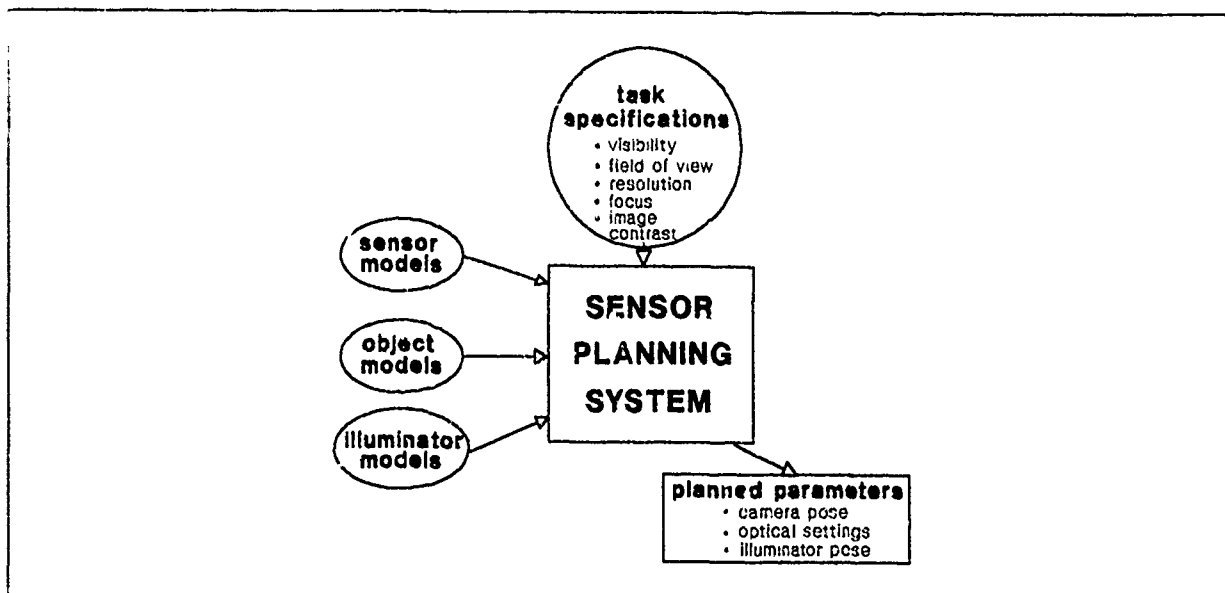


Figure 1. Model-based and task-driven sensor planning.

and sensor settings, when given as input:

- models of the environment (e.g. CAD models of parts, calibration models of the sensors) and
- the task requirements (e.g. feature resolution of 3 mls).

In currently employed sensor-based systems, such parameters are determined by a laborious and time-consuming trial and error approach, in which, sensor locations and settings are chosen and then tested to verify whether they meet the requirements of the task at hand. This procedure results in values that are valid for only a specific setup and that can potentially become unsatisfactory when errors (e.g. robot inaccuracy) alter the environment.

Consequently, these applications have limited intelligence and flexibility. This is the case despite the fact that there is information available that can be used to generate strategies to *plan* rather than search for many task parameters. For instance, the geometric information needed to automatically generate camera placement strategies in a part visual inspection and gaging application can be derived from the CAD/CAM models of these parts, that are often available today in manufacturing. In addition, the physical attributes needed for planning illumination, such as color and reflectivity of the part under inspection, could be included in the CAD database as well, so that the interaction between the light and the object surface can be properly modelled. On the other hand, camera and illuminator models embodying their physical properties (e.g. lens focal length, spatial distribution of light source irradiance) and geometrical properties (e.g. illumination angle of the light projector) can provide the planning system with the required sensor and illuminator characteristics.

The importance of the problem can be illustrated by the many advantages that will result if it is solved:

- the development cycle and cost of a sensor-based system is reduced as the parameters can be automatically determined (e.g. sensor emulation tool).
- sensor parameter values can be found that are robust, that is, satisfying the task requirements at hand even in the presence of uncertainty.
- sensor parameter values can be determined that optimize the sensor output with respect to a given task criterion. For instance in the case of vision sensing, camera and illuminator poses can be found that enhance the contrast between task relevant features and the background.
- the sensor-based system can become adaptive, automatically reconfiguring itself to accommodate variations in the workplace.

In general, this can lead to more intelligent sensor-based systems that would operate more flexibly and more autonomously as various processes can be planned and performed automatically and reliably.

2 APPROACH

We are developing a *model-based and task-driven vision system MVP*, (Machine Vision Planning), that automatically plans vision sensor parameters so that task requirements, common to most industrial machine vision applications, are satisfied.

More specifically, methods have been developed to determine camera poses and settings so that features of interest of polyhedral objects are:

- visible (occlusion-free positions of the sensor) [9, 12],
- contained entirely in the sensor field of view [11],
- in focus [7],
- resolvable by the sensor to a given specification [11].

Each task requirement generates an equivalent geometric constraint, which in turn is satisfied in a domain of admissible values in the space of parameters to be planned. These component admissible domains obtained for each task requirement are then intersected in order to determine parameter values that satisfy all constraints simultaneously.

In this paper we will describe the planning techniques developed for the task constraint of *feature resolution* and will briefly discuss how this constraint relates to the other feature detectability constraints mentioned above.

3 PAST WORK

The general sensor planning problem has received considerable interest recently and in this section we will briefly discuss and contrast related approaches to the problem. In particular, this review of past work will center mostly around the feature resolution constraint in particular. A more complete review can be found in [7].

Cowan and Kovesi [1, 2] considered the feature resolution constraint in their sensor planning work. They defined resolution to be the minimum angle (*resolution angle*) subtended by a given incremental surface length at the perspective center of the lens, rather than the minimum number of pixels per surface length which is commonly used. With this definition of resolution they determined the locus of the perspective center of the lens, when the given surface length is barely resolvable, to be the circular arc of points at which this surface length subtends a constant angle equal to the resolution angle. When placing the camera at viewpoints inside this circular arc locus, the incremental line-segment feature is resolvable, since it subtends an angle at the lens perspective center that is larger than the resolution angle. The viewing direction of the camera can be arbitrary given that, for any viewpoint inside this circular arc locus, the minimum resolution angle requirement is satisfied irrespective of the viewing direction. This circular arc locus was constructed by Cowan and Kovesi using an iterative procedure, since the resolution angle depends on the perspective center to image plane distance which is not known a priori. A direct method to obtain this resolution satisfying locus in closed-form has been developed in the MVP system as described in [11]. In addition, Cowan and Kovesi obtained the resolution satisfying domain for a planar facet feature as the intersection of the individual admissible domains for incremental circle features computed at points of the facet. The resolution satisfying locus for an incremental circle feature is the spherical arc generated by rotating the circular arc locus of an incremental line segment about the vertical axis through the midpoint of the line segment. Although the resolution definition that was employed can be shown to be more conservative than the one used customarily and, as described, reduces the sensor placement degrees of freedom from five (three positional and two rotational) to only the three positional, however, it is not the one used in practice for machine vision task specifications. In the MVP system the resolution constraint is formulated in terms of this more

established definition as will be seen in this paper (see section 4).

The VIO (Vision Illumination Object) system developed by R. Niepold and S. Sakane [4] planned the set-up of both a camera and a point light-source, given information regarding the environment as well as specifications of the task. The camera and illuminator positions however were both constrained to lie on the tessellated surface of a sphere centered at an object reference point and of a chosen radius. In addition, the camera optical axis was assumed to always point at this object reference point. The VIO system thus considered only two of the five (three positional and two rotational) *degrees of freedom for camera placement*. On the whole, VIO takes a *generate-and-test* approach to the problem. That is, sensor configurations are generated and then evaluated according to the task criteria. It is important to contrast this approach to the *synthesis* approach of SRI and MVP, where locations that satisfy the task are *directly* determined. VIO on the other hand first considers the combinatorial pairs of camera and illuminator locations, calculates an image representation of the expected scene for each pair, and then evaluates certain image feature attributes to assess the goodness of each such sensor configuration. Among the image feature attributes considered was *the length of the feature edge in the image*. That is, the VIO system essentially computed feature magnification for various camera viewpoints in a limited space of camera placement, rather than directly determining camera poses that satisfy a given resolution specification. HEAVEN is the precursor to the VIO system and incorporated earlier work by Sakane et al. [5, 6] in sensor planning in which feature resolution was not considered.

The ICE system was developed at the University of Washington [13, 14] to achieve automatic sensor and illumination planning for machine vision tasks. Similar to both VIO and HEAVEN, ICE takes a *generate-and-test* approach in a restricted space for sensor placement. For the most part the sensor planning problem is given little attention in the ICE system. The only sensor related task constraint considered in the sensor placement planning problem is *edge visibility*. That is, neither resolution nor magnification of the edges in the image was taken into account.

4 SENSOR PLANNING FOR THE FEATURE RESOLUTION CONSTRAINT

Customarily pixel resolution is used to indicate the approximate size of the smallest scene feature which can be seen by the vision system. In many machine vision tasks it is required that a particular unit feature size on an object appear as a minimum number of picture elements on a sensor. This feature resolution constraint can be satisfied by properly selecting the image sensor, as well as by carefully planning its placement and settings. The objective of sensor planning for the feature resolution constraint is to determine the sensor parameters that achieve this resolution.

In this section, we shall present a method to plan the camera pose and the optical settings of a lens, so that chosen features can be resolved to meet a given specification.

Consider the minimum feature A_iB_i of length l , shown in Figure 2, as well as its image of length w . The goal is to determine the locus of the perspective center O of the lens (or in general of the front nodal point, in the case of a thick lens) as described by $r = f_1(w, l, f, \gamma, \theta, \phi, \dots)$, as well as the perspective center to image plane distance $d = f_2(w, l, f, \gamma, \theta, \phi, \dots)$ that satisfy the resolution specification w/l where r, θ are the polar coordinates of the perspective center of the lens with respect to the coordinate system at A_i , ϕ is the angle between the optical axis Oz' and the plane defined by A_iB_i and O , γ is the angle between A_iB_i and the projection Oz'' of the optical axis onto the plane defined by A_iB_i and O , d is the effective focal length, namely the distance of the perspective center of the lens to the image plane, f is the intrinsic focal length of the lens, that is, the focal length of the lens for an object at infinity, l is the minimum feature to be resolved, w is the length of A_iB_i , the image of A_iB_i , and f_1, f_2 are the functional relationships to be determined.

As discussed in section 3, the existing approach to this problem [1, 2] involves an iterative procedure to estimate the object and image distances, D and d , that satisfy the resolution constraint. In [11] we presented a direct method to compute these distances, meeting the resolution requirement when the feature of interest is viewed orthogonally. In this paper we extend this work to determine the locus of camera poses and lens settings that satisfy the resolution constraint in a general viewing configuration.

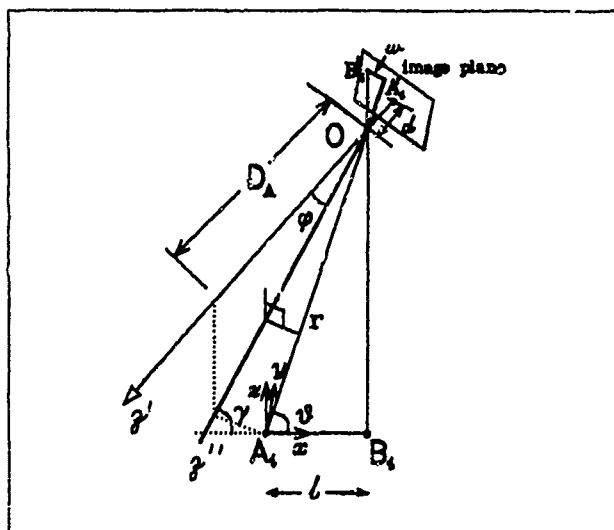


Figure 2. Perspective projection in 3D for a viewing orientation Oz' .

¹ the x axis is along A_iB_i and the y axis lies in the plane of O and A_iB_i .

4.1 The resolution constraint for orthogonal viewing

In Figure 3, when the feature AB is viewed orthogonally, then for every minimum feature A_iB_i of length l on AB that must satisfy the resolution specification, the following relationships hold:

$$\frac{1}{D} + \frac{1}{d} = \frac{1}{f} \quad (1a)$$

$$\frac{d}{D} = \frac{w}{l} \quad (1b)$$

Equation (1a) is the Gaussian lens law applied to a general coaxial optical system with the object and image distances, D and d , measured from the principal planes in the object and image spaces respectively. On the other hand, equation (1b) expresses the linear magnification of the minimum feature l . By combining (1a) and (1b), the object and image distances can be directly computed from the following relationships:

$$D = \left(1 + \frac{l}{w}\right)f \quad (2a)$$

$$d = \left(1 + \frac{w}{l}\right)f \quad (2b)$$

Consequently, given the resolution requirement w/l and the intrinsic focal length f of the lens, values for the object and image distances that satisfy the resolution constraint in the limit can be determined from (2a) and (2b) respectively. For

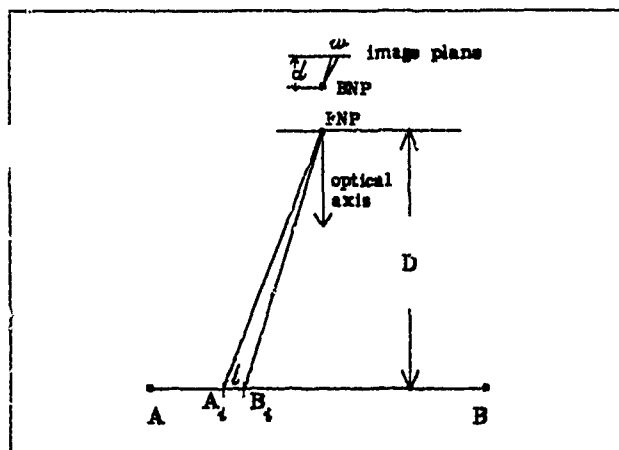


Figure 3. The resolution constraint for orthogonal viewing.

object distances smaller than the value of D computed from (2a) and their respective conjugate image distances, it is true that $d/D > w/l$ and therefore the resolution requirement is exceeded. In Figure 4, the resolution satisfying region for camera placement is shown shaded for an intrinsic focal length of $f = 4.4275$ in and the resolution specification of $w/l \approx 3$ frame buffer pixels per 0.01 in.

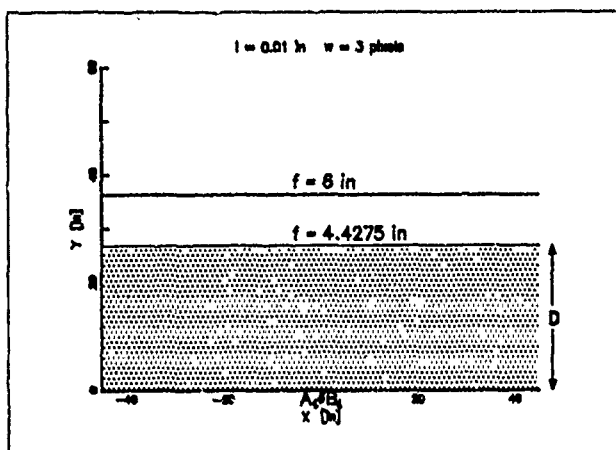


Figure 4. The resolution satisfying domain for orthogonal viewing.

It should be noted that in all the previous relationships, the length w of the image of the minimum feature is expressed in the sensor plane. However, when the limiting resolution is specified as the minimum number of picture elements in the frame buffer required to resolve l , then the scale factor [3] that relates the sensor element spacing of the CCD array to the pixel element spacing of the frame buffer must be known in order to compute w .

Previously for a constant intrinsic focal length lens, f is fixed and thus the object and image distances computed from (2a) and (2b) respectively, are unique. For a variable focal length lens (i.e. zoom lens), f varies and can be set as desired. In this case, D and d can assume a range of values from (2a) and (2b) limited only by the working range of the specific lens [11]. In Figure 4, the admissible domain for camera placement is shown to increase as the intrinsic focal length increases (i.e. the lens is "zoomed-in").

4.2 The resolution constraint for a general viewing direction in 2D

In the previous section the resolution constraint was discussed in the case of orthogonal viewing. In other words, camera locations were determined for which, a given minimum feature size in object space is projected to a certain number of pixels in the image, when the optical axis is perpendicular to the feature. However in a general viewing situation the angle between the optical axis and a line-segment feature may take on values other than $\pi/2$ in which case perspective foreshortening is nonuniform. As a result, in order to control the image size of a minimum feature dimension in object space, when in general the feature is viewed at angle, it is important to understand how the perspective transformation of a given length changes as the location of the perspective center and the direction of the optical axis vary.

Let us consider the two-dimensional case shown in Figure 5, where the optical axis lies on the plane defined by

the line-segment feature $A_i B_i$ and the perspective center O and makes an angle γ with $A_i B_i$ (for simplicity the lens perspective center has been used rather than the front and back nodal points of the lens, however all relationships hold for the case of a thick lens as well). The coordinate system is taken at A_i with the x -axis along $A_i B_i$, while points are expressed in polar coordinates. The problem then involves determining the locus of the perspective center O , such that the projection $A' B'$ of the feature $A_i B_i$ onto the image plane has length w .

Initially let us assume that the distance d from the image plane to the perspective center is known a priori. The case where d varies, as the lens is focused at different settings depending on the object distance, will be addressed later. It can be shown that for the case of a constant d , the locus of the perspective center, such that the image feature has length w , is described in polar coordinates by the following relationship:

$$r = \frac{dl \sin \theta}{w \cos^2(\theta - \gamma)} + \frac{l \cos \gamma}{\cos(\theta - \gamma)} \quad (3)$$

where (r, θ) are the polar coordinates of the perspective center O , $0 \leq \theta \leq \pi$, l is the length of the feature line-segment, w is the length of the feature line-segment in the image and γ is the angle between the optical axis and the line-segment. This angle γ is taken in the range $[0, \pi/2]$, since the case where $\pi/2 \leq \gamma \leq \pi$ is symmetrical with the axis of symmetry

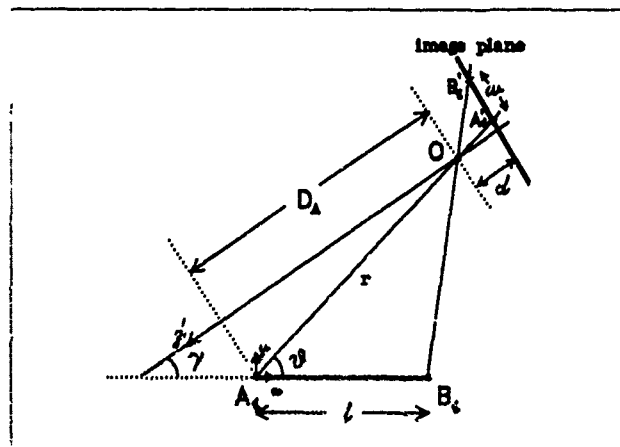


Figure 5. Perspective projection in 2D for a viewing orientation $O\gamma'$.

being the perpendicular bisector of $A_i B_i$. A polar plot of this relationship, when $\gamma = \pi/4$, $l = 0.1$ in, $w = 6$ pixels, is shown in Figure 6. Thus, when viewing at a given angle γ , the locus of viewpoints, for which the length of the line-segment in the image is at least w , is the region shown shaded in Figure 6. This region is bounded by the curve described by (3) and the line which is perpendicular to the optical axis direction and passes through the point of $A_i B_i$ that is closest to the perspective center in the optical axis direction. (i.e. B_i in this case). The latter line ensures that, for this viewing direction γ , all points of $A_i B_i$ can be imaged.

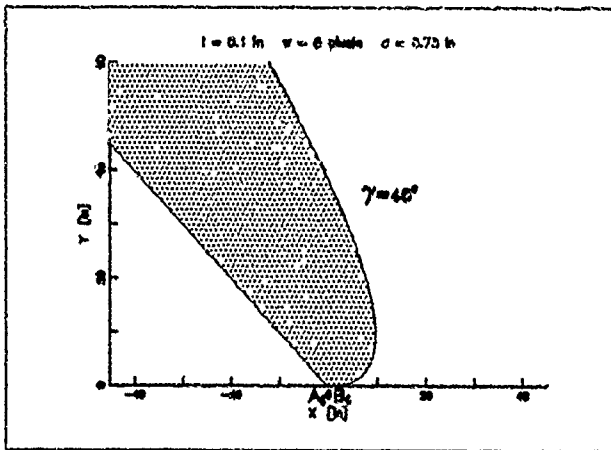


Figure 6. The locus of the perspective center for which the feature AB has an image of length w when viewed at a 45 degree angle.

Similarly, the locus of viewpoints that satisfy this resolution specification, when viewing the line-segment feature at any angle in the range (γ_1, γ_2) , is the region bounded by the envelope,² E_{γ_1, γ_2} , of the family of curves described by (3), where $\gamma_1 < \gamma < \gamma_2$, and the line which is perpendicular to the optical axis direction for the smallest viewing angle γ (i.e. γ_1), and which passes through the point of $A_i B_i$ closest to the perspective center in the optical axis direction (i.e. B_i in this case). The envelope E_{γ_1, γ_2} for $\pi/4 < \gamma < \pi/3$, and the associated admissible region are shown in Figure 7 for a particular resolution specification. It can be observed that the envelope E_{γ_1, γ_2} consists of a segment of the curve given by (3) for $\gamma = \gamma_1$, a similar segment for $\gamma = \gamma_2$, and a segment joining the previous two, which is a circular arc, as will be explained in the next paragraph. With this information it is possible to construct this envelope analytically.

It is interesting to note that the admissible region for all possible angles γ is bounded by a circular arc, as illustrated in Figure 8. This circular arc can be determined analytically, since, it can be shown to be the locus of points at which the line-segment feature subtends a constant angle $\delta = 2 \tan^{-1}(w/2d)$, where d is given by (2b). The radius of this circular arc is $R = l/(2 \sin \delta)$. This locus was used by Cowan in [2] and was determined using an iterative technique (see section 3). It is clear that this circular arc is in general a conservative estimate of the admissible domain of camera viewpoints.

Let us now turn to the case where the image distance d varies, so that the feature line-segment is in focus at all times. In the following analysis, the lens will be focused at the point of the line-segment feature that is farthest in distance along the direction of the optical axis, that is, point A , when $0 \leq \gamma \leq \pi/2$. A similar procedure can be applied when the lens is focused at any other chosen point. In this situation, the image distance d can be expressed in terms of the

² an envelope of a family of curves is a curve that is tangent at each of its points to some curve of the family.

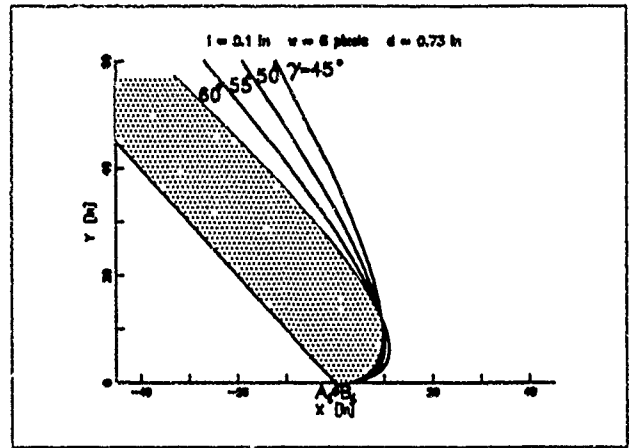


Figure 7. The resolution constraint when viewing at any angle between 45 and 60 degrees.

object distance D_A and the lens focal-length f using the Gaussian lens law:

$$d = \frac{D_A f}{D_A + f} \quad (4)$$

where

$$D_A = r \cos(\theta - \gamma) \quad (5)$$

The above relationships can be combined with (3) to arrive at a second-order polynomial in D_A , which can be solved and hence r and d can be determined by using (4) and (5):

$$r = \frac{wf \cos(\theta - \gamma) + lw \cos \gamma \cos(\theta - \gamma) + lf \sin \theta + \sqrt{\Delta}}{2w \cos(\theta - \gamma)} \quad (6)$$

$$\Delta = [wf \cos(\theta - \gamma) + lw \cos \gamma \cos(\theta - \gamma) + lf \sin \theta]^2 - 4w^2 lf \cos \gamma \cos^2(\theta - \gamma) \quad (7)$$

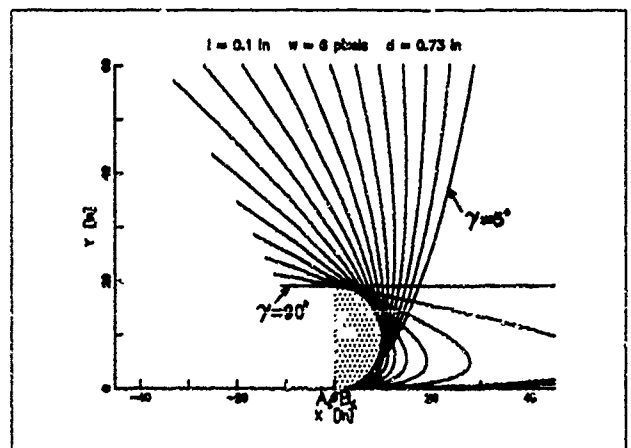


Figure 8. The resolution constraint when viewing at any angle between 0 and 90 degrees.

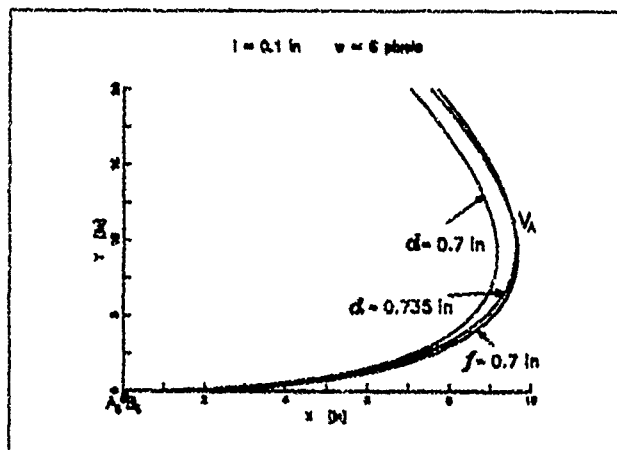


Figure 9. The resolution constraint for a variable image distance d when viewing at a 45 degree angle.

Thus, for the case where the image distance varies to achieve proper focusing, the above formulas give a closed-form solution for the polar distance r of the perspective center O , as well as the image distance d , so that the image of the line-segment feature has a specified length. In Figure 9, a polar plot of this relationship, when $\gamma = \pi/4$, $l = 0.1$ in, $w = 6$ pixels $f = 0.7$ in, is shown overlapped with the plot of (3) for $d = 0.7$ in and $d = 0.735$ in, and the same input otherwise. It can be seen that the approximation $d = f$ is conservative and it may be considered acceptable in cases where the working range of the image distance d is small.

4.3 The resolution constraint for a general viewing direction in 3D

Let us now consider the three-dimensional viewing situation shown in Figure 2, where the optical axis makes an angle ϕ with the plane defined by A, B , and O , while the projection of the optical axis onto this plane makes an angle γ with A, B . With a similar analysis it can be shown that, for the 3D case and a known image distance d , the locus of the perspective center, such that the image feature has length w , is described in polar coordinates by the following relationship:

$$r = \frac{dl \sin \theta}{w \cos \phi \cos^2(\theta - \gamma)} + \frac{l \cos \gamma}{\cos(\theta - \gamma)} \quad (6)$$

The above equation is identical to (3) except for the extra $\cos \phi$ in the denominator of the first term. As a result, the admissible region increases with ϕ , since the feature draws closer in the direction of the optical axis (see Figure 10).

When the image distance varies to achieve proper focusing, the polar distance r of the lens perspective center and the image distance d can be found, in a manner similar to the 2D case by combining (6) and the following relationships:

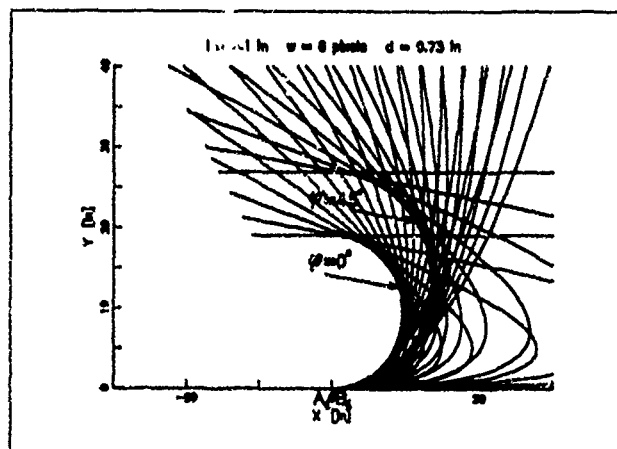


Figure 10. The resolution constraint in 3D.

$$d = \frac{D_A f}{D_A + f}$$

where $D_A = r \cos(\theta - \gamma) \cos \phi$.

4.4 Resolution "vs." Magnification

In effect, all previous analyses considered how to achieve a particular level of feature *magnification*. However, these results, when applied locally, can be used to ensure that each minimum feature length can be resolved (i.e. covered by distinct pixels). For instance, let AB in Figure 11 be the feature that must be resolved to 15 mils. The admissible region for camera placement, that satisfies this resolution constraint for AB as a whole, is determined by the corresponding region for a 15 mil mini-feature on AB that is farthest from the perspective center along the optical axis. For instance, when $0 \leq \gamma \leq \pi/2$ this mini-feature is l_A , since, as shown in Figure 11, all other mini-features of this size, such as l_B , are redundant. Similarly, when $\pi/2 \leq \gamma \leq \pi$, it is l_B that dictates the resolution constraint for AB as a whole. On the other hand, the locus of camera viewpoints for which the entire feature satisfies an equivalent *magnification* (i.e. 1/0.015 pixels per inch) is shown in Figure 11 to lie, as expected, between the two mini-feature resolution loci.

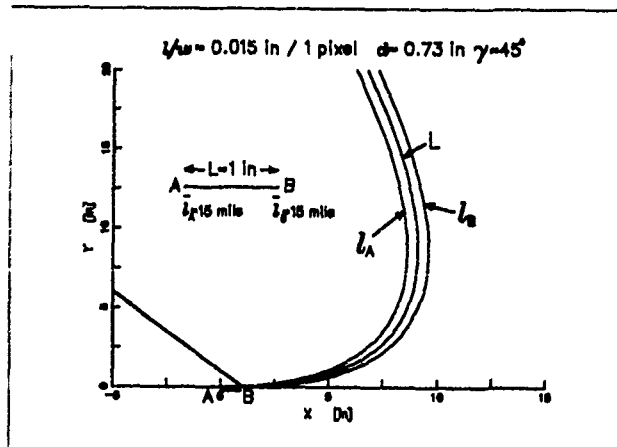


Figure 11. Resolution vs. magnification.

5 EXPERIMENTS

As part of our MVP system, we have developed and implemented machine vision planning algorithms for the feature detectability constraints of visibility, resolution and field of view. In the experiments, we demonstrate the effectiveness of this approach using a robot vision system that plans its pose and the lens settings of its camera according to these techniques. In this section, we present test results demonstrating satisfaction of all three feature detectability constraints and we discuss the resolution planning technique in detail.

5.1 Setup Description

The experimental setup is shown in Figure 12. A Javelin CCD 480 x 384 camera is fastened to the last joint of an IBM Clean Room Robot (CRR). The CRR has two manipulators, each with seven joints, which consist of three linear joints (x, y, z), three rotary joints (roll, pitch and yaw) and the gripper joint.

A Vicon zoom lens with two close-up lenses or diopters (4-diopter and 2-diopter, making it a 6-diopter) is mounted on the CCD camera. The zoom lens has three motorized functions: zoom, focus and iris. For zoom and focus, potentiometers provide feedback of the lens element position. The zoom ratio of the lens is 6X with a focal length range of 12.5-75 mm (without the 6 diopter close-up lenses).

The object used in the camera placement experiments is shown in Figure 13 and a CAD model of it is shown in Figure 14. The feature to be viewed is the top face T of the enclosed cube. This object is assembled from smaller primitive objects (i.e. cubes, parallelepipeds etc.) so that it can be reconfigured to test a variety of occlusion arrangements.



Figure 12. The experimental setup.

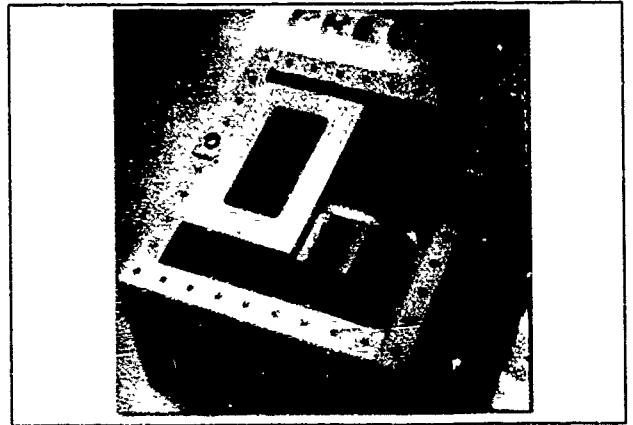


Figure 13. The object.

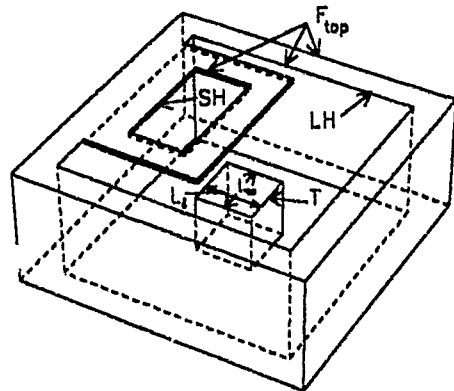


Figure 14. CAD model of the object.

5.2 Results

The domain of feasible camera locations is initially limited to the region in three-dimensional space from where the feature to be observed is *visible*. This region is generated by the visibility algorithm that we have developed [9, 12]. For the object and target shown in Figure 14, the visibility region was determined and, as shown in Figure 15, it consists of two disjoint visibility volumes that correspond to viewing the target through the small hole SH and the large hole LH of the object (Figure 14).

Viewpoints chosen from the visibility region must also satisfy the field of view constraint, that is, the target must not be truncated in the image as a result of the finite size of the sensor plane. As a result, given the target and the dimensions of the sensor plane, this constraint generates its own admissible region [11]. When placing the lens center inside this region, there exist orientations of the optical axis (e.g. range of values for the angles γ and ϕ) that maintain the target within the field of view.

5 EXPERIMENTS

As part of our MVP system, we have developed and implemented machine vision planning algorithms for the feature detectability constraints of visibility, resolution and field of view. In the experiments, we demonstrate the effectiveness of this approach using a robot vision system that plans its pose and the lens settings of its camera according to these techniques. In this section, we present test results demonstrating satisfaction of all three feature detectability constraints and we discuss the resolution planning technique in detail.

5.1 Setup Description

The experimental setup is shown in Figure 12. A Javelin CCD 480 x 384 camera is fastened to the last joint of an IBM Clean Room Robot (CRR). The CRR has two manipulators, each with seven joints, which consist of three linear joints (x, y, z), three rotary joints (roll, pitch and yaw) and the gripper joint.

A Vicon zoom lens with two close-up lenses or diopters (4-diopter and 2-diopter, making it a 6-diopter) is mounted on the CCD camera. The zoom lens has three motorized functions: zoom, focus and iris. For zoom and focus, potentiometers provide feedback of the lens element position. The zoom ratio of the lens is 6X with a focal length range of 12.5-75 mm (without the 6 diopter close-up lenses).

The object used in the camera placement experiments is shown in Figure 13 and a CAD model of it is shown in Figure 14. The feature to be viewed is the top face T of the enclosed cube. This object is assembled from smaller primitive objects (i.e. cubes, parallelepipeds etc.) so that it can be reconfigured to test a variety of occlusion arrangements.



Figure 12. The experimental setup.

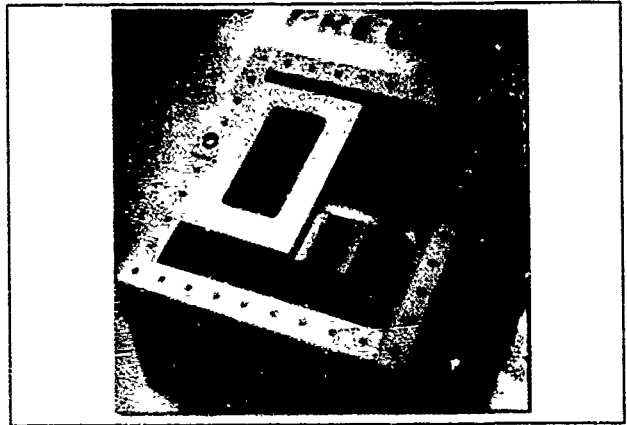


Figure 13. The object.

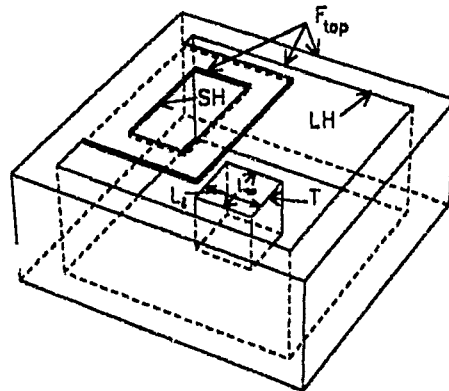


Figure 14. CAD model of the object.

5.2 Results

The domain of feasible camera locations is initially limited to the region in three-dimensional space from where the feature to be observed is *visible*. This region is generated by the visibility algorithm that we have developed [9, 12]. For the object and target shown in Figure 14, the visibility region was determined and, as shown in Figure 15, it consists of two disjoint visibility volumes that correspond to viewing the target through the small hole SH and the large hole LH of the object (Figure 14).

Viewpoints chosen from the visibility region must also satisfy the field of view constraint, that is, the target must not be truncated in the image as a result of the finite size of the sensor plane. As a result, given the target and the dimensions of the sensor plane, this constraint generates its own admissible region [11]. When placing the lens center inside this region, there exist orientations of the optical axis (e.g. range of values for the angles γ and ϕ) that maintain the target within the field of view.

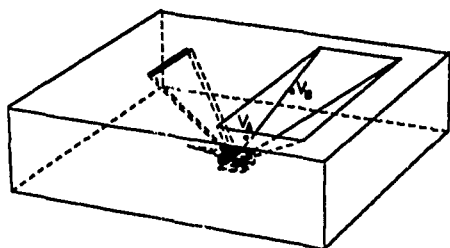


Figure 15. The visibility regions.

5.2.1 Resolution planning results for orthogonal viewing

The feature of interest that is chosen to satisfy the resolution constraint, for the case of orthogonal viewing, is the width of the target T , shown as L_w in Figure 14, where $L_w = 1$ in. The resolution limit is taken to be 3 frame buffer pixels per 0.01 inches. For this resolution limit and for the case of viewing this feature orthogonally, a viewpoint is found that just barely satisfies this resolution. That is, each image pixel spacing corresponds to 1/300 inches in object space, and thus the whole 1 inch width of the target projects to 300 pixels.

For the case of orthogonal viewing, object and image distances can be found from (2a) and (2b) for different intrinsic focal lengths of the zoom lens. As shown in Figure 4, for an intrinsic focal length of $f = 4.4275$ in:

$$D = (1 + \frac{l}{w})f = (1 + 1/0.192) \times 4.4275 = 26.74 \text{ in}$$

$$d = (1 + \frac{w}{l})f = (1 + 0.192) \times 4.4275 = 5.278 \text{ in}$$

where

$$\frac{w}{l} = \frac{3 \times 23 \times 0.70642}{0.01 \times 25.4 \times 1000} = 0.192$$

$w = 3$ pixels, $l = 0.01$ inches, 23 microns is the sensor element spacing in the horizontal direction and 0.70642 is the horizontal scale factor relating the sensor element spacing to the pixel spacing in the frame buffer.

Having determined the image distance, the lens can be focused accordingly. In addition, knowing that the optical axis orientation is perpendicular to the feature and having determined the object distance, the camera is then placed inside both the visibility and field of view satisfying regions.

5.2.2 Resolution planning results for non-orthogonal viewing

In this case, the feature of interest is chosen to be the length of the target T , shown as L_l in Figure 14, where $L_l = 1$ in. The resolution limit is taken to be 6 pixels per 0.1 inches specified in the image frame buffer. For this resolution

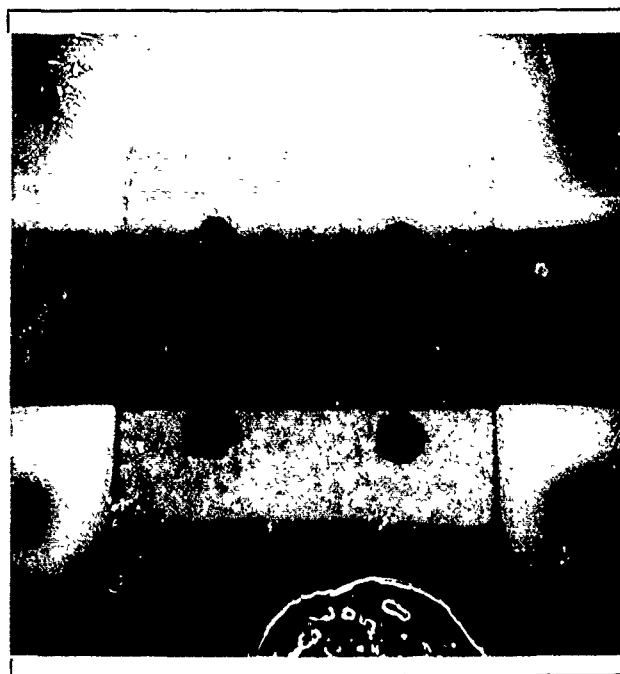


Figure 16. The view of the target for the case of orthogonal viewing.

specification, a viewpoint is found that again just barely satisfies this resolution. That is the viewpoint lies on the locus described by (6) and (7) and inside the visibility and field of view satisfying regions. The viewpoint chosen is shown in Figure 9 as V_A , for which $\gamma = 45^\circ$, $\theta = 50^\circ$, $l = 0.1$ in, $w = 6$ pixels and $f = 0.7$ in. For this viewpoint, r and d are found from (4), (5), (6) and (7) to be: $r = 14.75$ in, $d = 0.735$ in.

Knowing the image distance, the lens can be focused accordingly, and having determined the polar distance and angle of the lens center, as well as the optical axis orientation, the camera can also be placed.

5.2.3 Camera placement, lens setting and verification of the sensor planning results.

These two camera locations are shown in Figure 15 as viewpoints V_A for the case of non-orthogonal viewing and V_B for the orthogonal viewing case. The manipulator with the mounted camera is used to place the camera at the chosen viewing positions with respect to the occluding object and target. Each camera position chosen is known only with respect to an object coordinate system. What needs to be determined is the manipulator location that places the camera at the chosen position. This manipulator location can be computed from the hand-eye relationship [10] and the pose of the object in the robot world coordinate system [8]. Figure 12 shows the manipulator placed at viewpoint V_B and oriented orthogonally to the feature.

The associated scenes of the target from these viewpoints are shown in Figure 16 and Figure 17. These figures

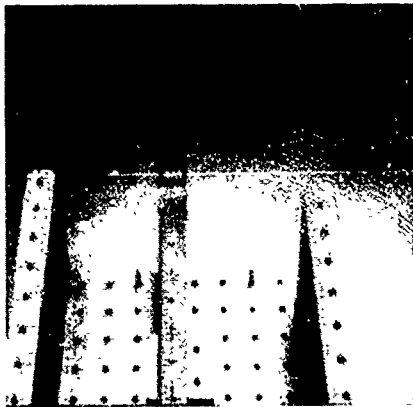


Figure 17. The view of the target for the case of non-orthogonal viewing.

contain an engraved steel ruler with graduations of 0.1 and 0.01 of an inch so that the resolution can be verified. The three small bright line segments in Figure 16 are approximately 6 frame buffer pixels and can be seen on the ruler to correspond to 0.02 inches at the far right, in the middle and at the far left of the feature. For the case of non-orthogonal viewing, shown in Figure 17, the farthest 0.1 inch feature appears as 5.6 ± 0.5 pixels. This measurement was done using threshold based edge finding using the ruler graduations as the edge features (Figure 18). The discrepancy between the resolution measured and that specified is, for the most part, due to the placement error of the robot manipulator.

6 CONCLUSION

We presented techniques to determine the placement and optical settings of a camera, so that a given resolution requirement is satisfied for chosen object features. The method generates the complete locus of camera poses and settings in an analytic manner. This approach improves on the currently employed techniques in which sensor configurations are generated and then tested for satisfaction of the task requirements. The results are valid for a general three-dimensional viewing configuration and were demonstrated using a robot vision system.

The results discussed in this paper, as well as those in [7, 9, 11, 12], are useful for automating the vision system design process, as well as for programming the vision system itself. In addition, such planning techniques will also prove useful for automated robot imaging systems that will be able to reconfigure themselves in an intelligent manner in order to optimize imaging quality.

* this author was supported in part by DARPA contract N00039-84-C-0165 and in part by Manufacturing Research, IBM T.J. Watson Research Center.

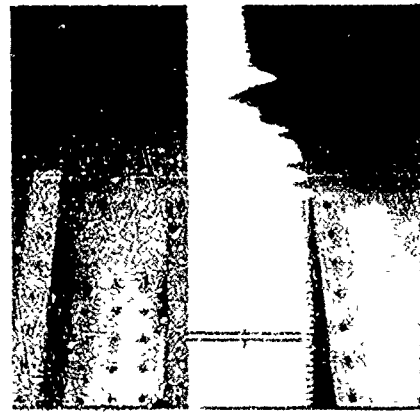


Figure 18. Measuring the resolution.

7 BIBLIOGRAPHY

- [1] Cowan, C. K., "Model based synthesis of sensor location," *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pp. 900-5, 1988.
- [2] Cowan, C. K., and Kovesi, P. D., "Automatic sensor placement from vision task requirements," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, pp. 407-16, May 1988.
- [3] Lenz, R. K., and Tsai, R. Y., "Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, pp. 713-20, Sept. 1988.
- [4] Niepold, R., and Sakane, S., "VIO - an approach to vision sensor and illumination planning using environmental models," *Computer Vision, Graphics, and Image Processing (submitted)*, November 1988.
- [5] Sakane, S., Ishii, M., and Kakikura, M., "Occlusion avoidance of visual sensors based on a hand eye action simulator system: HEAVEN," *Adv. Robot.*, vol. 2, pp. 149-65, 1987.
- [6] Sakane, S., Sato, T., and Kakikura, M., "Planning focus of attentions for visual feedback control," *Trans. Soc. Instrum. Control Eng.*, vol. 24, pp. 608-15, June 1988.
- [7] Tarabanis, K., *Model-Based and Task-Driven Machine Vision Planning*, New York, NY: Computer Science Department, Columbia University, CUCS-028-90, May 1990.
- [8] Tarabanis, K., and Tsai, R. Y., *Camera placement planning avoiding occlusion: Test results using a robotic hand/eye system.*, New York, NY: Computer Science Department, Columbia University, CUCS-501-89, October 1989.

- [9] Tarabanis, K., and Tsai, R. Y., "Viewpoint planning: the visibility constraint," *Proc. DARPA Image Understanding Workshop*, Palo Alto, California, May 23-26 1989.
- [10] Tsai, R. Y., and Lenz, R. K., "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration," *IEEE Trans. Robot. Autom.*, vol. 5, pp. 345-58, June 1989.
- [11] Tsai, R. Y., and Tarabanis, K., "Model-based planning of sensor placement and optical settings," *Proc. Sensor Fusion II: Human and Machine Strategies*, Philadelphia, Pennsylvania, November 6-9 1989.
- [12] Tsai, R. Y., and Tarabanis, K., "Occlusion-free sensor placement planning," in Freeman, H., editor, *Machine Vision for Three-Dimensional Scenes*, San Diego, California: Academic Press, 1990.
- [13] Yi, S., Haralick, R. M., and Shapiro, L. G., Automatic sensor and light source positioning for machine vision, Seattle, WA: University of Washington, EE-ISL-89-04, September 1989.
- [14] Yi, S., Haralick, R. M., and Shapiro, L. G., An illumination model for machine vision, Seattle, WA: University of Washington, EE-ISL-89-03, September 1989.

Statistical Decision Theory for Sensor Fusion

Raymond McKendall*

GRASP Laboratory

Department of Computer and Information Science

University of Pennsylvania

Philadelphia, PA 19104-6389

Abstract

This article is a brief introduction to statistical decision theory. It provides background for understanding the research problems in decision theory motivated by the sensor-fusion problem.

1 Introduction

This article is a brief introduction to statistical decision theory. It provides background for understanding the research problems in decision theory motivated by the sensor fusion problem. In particular, this article is an introduction for the articles *Robust Multi-Sensor Fusion: A Decision Theoretic Approach* [Kamberova and Mintz, 1990] and *Non-Monotonic Decision Rules for Sensor Fusion* [McKendall and Mintz, 1990] of these Proceedings. The principal references for this review are [Ferguson, 1967], [Berger, 1985], and [DeGroot, 1970]. The appendices of [McKendall, 1990] give an expanded discussion of sensor fusion and an expanded introduction to decision theory.

Section 2 states the general research problem in statistical estimation. Section 3 describes the sensor fusion problem. Section 4 introduces statistical decision theory and formulates the research problem as a decision problem. Section 5 gives some examples of the research problems studied.

2 Research Problem

The problem of this research to estimate the location parameter $\theta \in \Theta$ of a single observation Z in the model

$$Z = \theta + V.$$

Thus, the random variable Z is a measurement of the location parameter θ in continuous, additive noise V . The goal of this research is to estimate θ . The tool for analysis is statistical decision theory.

There are two versions of this problem, standard estimation and robust estimation. In a *standard-estimation* problem, the distribution function F of the additive noise V is known. An example is to estimate the mean θ of

$Z \sim \mathcal{N}(\theta, 1)$; in this case $F \sim \mathcal{N}(0, 1)$. (The notation $\mathcal{N}(\mu, \sigma^2)$ indicates a normal or Gaussian distribution with location μ and scale σ .) In a *robust-estimation* problem, the distribution F is uncertain: It is an unknown member of a given class \mathcal{F} of distribution functions, an uncertainty class. An example is to estimate the mean θ of $Z \sim \mathcal{N}(\theta, \sigma^2)$ when $\sigma \in (0, 1]$ is unknown; in this case $F \in \mathcal{F}$ where \mathcal{F} is the set of $\mathcal{N}(0, \sigma^2)$ distribution functions with $\sigma \in (0, 1]$. Robust estimation accounts for inexact characterizations of the noise. Many problems in robust estimation reduce to problems in standard estimation.

The statement of the problem in terms of mathematical statistics is to estimate the location parameter $\theta \in \Theta$ of the random variable Z , where

$$Z \sim F_Z(\cdot|\theta)$$

and

$$F_Z(z|\theta) = F(z - \theta), \quad \forall z \in \mathcal{R}.$$

The distribution $F_Z(\cdot|\theta)$ of Z is the *sampling distribution*, and the distribution F of V is the *nominal distribution*. Similarly, the density of Z , $f_Z(\cdot|\theta)$, is the *sampling density*, and the density f of V is the *nominal density*. The density functions are related by the equation

$$f_Z(z|\theta) = f(z - \theta), \quad \forall z \in \mathcal{R}.$$

3 Motivation

The location-estimation model of this research is fundamental to research in robust fusion of location data. *Location data* are sensors' measurements of the position of an object. *Fusion* is the combination of location data from different sensors. *Robust fusion* accounts for uncertainty in the description of the underlying system. The goals of the research in sensor fusion are to model sensor fusion as a statistical problem, to analyze the model with statistical decision theory, and to develop mathematical statistics for the analysis.

Example 1 illustrates a sensor-fusion problem with three sensors. The sensors S_1 , S_2 , and S_3 may be different kinds of sensors. For example, S_1 may be a laser sensor, S_2 may be a sonar sensor, and S_3 may be a camera. The output of each sensor S_i is a measurement Z_i of the distance θ of the object T from the horizontal axis. The dashed box around each sensor represents

*Acknowledgement: Navy Contract N0014-88-K-0630; AFOSR Grants 88-0244, 88-0296; Army/DAAL 03-89-C-0031PRI; NSF Grants CISE/CDA 88-22719, IRI 89-06770; and the Dupont Corporation.

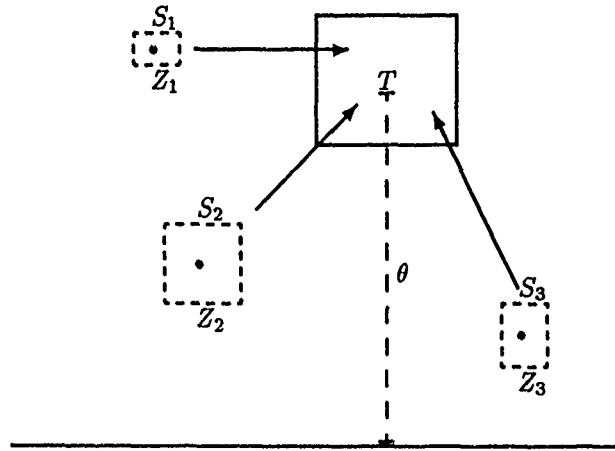


Figure 1: Sensor-fusion paradigm

the noise associated with the sensor's measurement. For example, there may be uncertainty in the exact position of each sensor. The box around the object T represents the prior information about the location of the object. For example, the object may be in a room with known dimensions.

The fusion problem is to combine the three measurements Z_1 , Z_2 , and Z_3 of the distance θ into a single estimate. Fusion of the data requires that the data are consistent: The consistency problem is to verify that Z_1 , Z_2 , and Z_3 are measurements of the same parameter. \square

The location-data paradigm consists of a measurement Z of an unknown parameter θ in statistical uncertainty, noise due to the environment or to the sensor itself. A location model of a measurement assumes that the parameter governs only the location of the noise but not its shape; the model assumes that the shape of the noise is independent of the parameter. (Such noise is called additive.) For example, a measurement Z of a parameter θ may be modeled as a normally distributed random variable with mean θ : $Z \sim \mathcal{N}(\theta, \sigma^2)$. Then the shape of the noise is $\mathcal{N}(0, \sigma^2)$ regardless of the location θ of the mean.

The sensor-fusion problem has multiple measurements Z_1, \dots, Z_n of the location θ in additive noise. These measurements originate from different sensors. The fusion problem is to combine these data into a single value for the location θ . The model assumes a tolerance ϵ for error: An estimate $\hat{\theta}$ for θ is acceptable if the absolute error of estimation $|\hat{\theta} - \theta|$ is at most ϵ ; otherwise, the error is unacceptable. The goal of fusion is to find an estimator that minimizes the probability of unacceptable error. The fusion problem subsumes the problem of consistency, which is to verify that the data Z_1, \dots, Z_n are in fact measurements of the same location.

4 Decision Theory

The tool for the analysis of the location-estimation problem is statistical decision theory. This section introduces

this theory and formulates the location-estimation model as a decision problem.

The Decision Problem

Figure 2 illustrates the structure of a statistical decision problem. The task is to make a decision or perform some action a from a set \mathcal{A} of allowable actions. The parameter ω determines the correct action to take, but the value of this parameter is not known. There are, however, two types of information about ω . First, the possible values are known. These are the elements of the set Ω . Second, there is an observable random variable Z whose distribution depends on ω and thus contains statistical information about ω . The goal of a decision problem is to choose an action from \mathcal{A} by using the observable to gain information about the unknown parameter. The objective is to find a *decision rule* δ that maps the sample space \mathcal{Z} of the observable Z to the action space \mathcal{A} : The decision or action for an observation $Z = z$ is $\delta(z) \in \mathcal{A}$. Because the action taken is based on a random variable, the decision process has error. The loss function L gives the penalty for this error: The loss incurred by action a for the parameter ω is $L(\omega, a)$.

In summary, a decision problem is a quadruple $(\Omega, \mathcal{A}, L, Z)$ consisting of a parameter space Ω , an action space \mathcal{A} , a loss function L , and an observable Z . The *parameter space* is the set of possible values for the unknown statistical parameters. For standard estimation, the parameter space is $\Omega = \Theta$. For robust estimation, the parameter space is $\Omega = \Theta \times \mathcal{F}$. The *action space* is the set of available decisions. The action space of the location-estimation problem is $\mathcal{A} = \Theta$; an action $a \in \mathcal{A}$ is an estimate of θ . The *loss function* is a scalar function on $\Omega \times \mathcal{A}$. The loss $L(\omega, a)$ for $\omega \in \Omega$ is the cost of the estimate a of θ . This research uses the *zero-one (ϵ) loss function*, L_ϵ :

$$L_\epsilon(\omega, a) := \begin{cases} 0 & \text{if } |\theta - a| \leq \epsilon \\ 1 & \text{if } |\theta - a| > \epsilon \end{cases}$$

The *observable* is a random variable whose distribution depends on the unknown parameters and thus contains

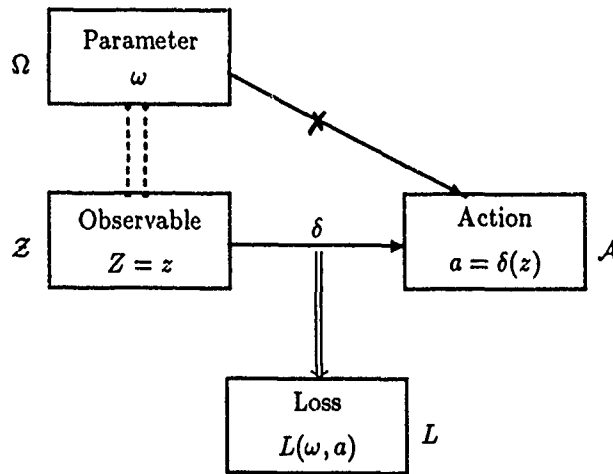


Figure 2: A statistical decision problem

information about them. For the location-estimation problem, the observable is $Z = \theta + V$.

A decision rule $\delta(Z)$ in an estimation problem is an estimator of θ . The decision rule is chosen according to an optimality criterion. This research constructs minimax decision rules: Under zero-one (e) loss, an estimator $\delta^*(Z)$ of the location parameter θ is minimax if

$$\sup_{\omega} P_{\omega}\{|\delta^*(Z) - \theta| > e\} = \inf_{\delta} \sup_{\omega} P_{\omega}\{|\delta(Z) - \theta| > e\}.$$

Thus, a minimax estimator based on zero-one (e) loss minimizes the maximum probability that the absolute error of estimation is greater than e . Equivalently, this estimator minimizes the maximum probability of unacceptable error.

Optimal Decision Rules

A decision rule δ_1 is preferable to a decision rule δ_2 if the loss under δ_1 is smaller than the loss under δ_2 . The loss function alone, however, is not enough to choose between two decision rules since $L(\omega, \delta(Z))$ is a random variable. Thus, the first step in evaluating the performance of a decision rule δ is to find its average loss or *risk* $R(\omega, \delta)$:

$$\begin{aligned} R(\omega, \delta) &:= E[L(\omega, \delta(Z))] \\ &= \int_{\mathcal{Z}} L(\omega, \delta(z)) dF_Z(z|\theta) \end{aligned}$$

The risk $R(\omega, \delta)$ is the weighted-average loss of δ , where the weight is given by the distribution $F_Z(\cdot|\theta)$.

Example When the loss is zero-one (e), the risk of a rule δ is the probability under ω that the absolute error exceeds e :

$$\begin{aligned} R(\omega, \delta) &= \int_{\mathcal{Z}} L_e(\theta, \delta(z)) dF_Z(z|\theta) \\ &= \int_{\{z: |\delta(z) - \theta| > e\}} dF_Z(z|\theta) \\ &= P_{\omega}\{|\delta(Z) - \theta| > e\} \end{aligned}$$

Thus, small risk implies small probability of unacceptable error of estimation. \square

Comparison of risk gives a weak optimality criterion. A decision rule δ_1 is preferable to a decision rule δ_2 if the risk of δ_1 is smaller than the risk of δ_2 uniformly in ω . A decision rule is *admissible* if there is no other rule preferable to it. Comparison of risk, however, is an incomplete criterion since the risk varies in the unknown parameter ω . (See figure 3.) Thus, the second step in finding a decision rule is to remove the dependence of a choice on the unknown parameter. This step leads to three types of decision rules: minimax, Bayes, and equalizer.

The minimax approach eliminates the unknown parameter ω from the risk by comparing the maximum risks of two decision rules. A decision rule δ^* is a *minimax* rule if its maximum risk is the smallest possible maximum risk:

$$\sup_{\omega} R(\omega, \delta^*) = \inf_{\delta} \sup_{\omega} R(\omega, \delta)$$

Thus, a minimax rule guards against the worst-possible risk.

The Bayes approach eliminates ω by comparing the weighted-average risks of two decision rules. This approach assumes that there is a known probability distribution π on the parameter space Ω through which the risks are averaged. This distribution is the *prior distribution* on Ω . A decision rule δ^* is *Bayes* against π if its weighted-average risk under π is the smallest possible weighted-average risk:

$$E[R(\omega, \delta^*)] = \inf_{\delta} E[R(\omega, \delta)]$$

Thus, a Bayes rule guards against the worst-possible weighted-average risk.

The equalizer approach eliminates ω by choosing a decision rule with constant risk. A decision rule δ is an *equalizer* rule if for all $\omega \in \Omega$,

$$R(\omega, \delta) = \text{constant}.$$

The goal of this research is to find a minimax rule for the location parameter θ of the measurement $Z = \theta + V$,

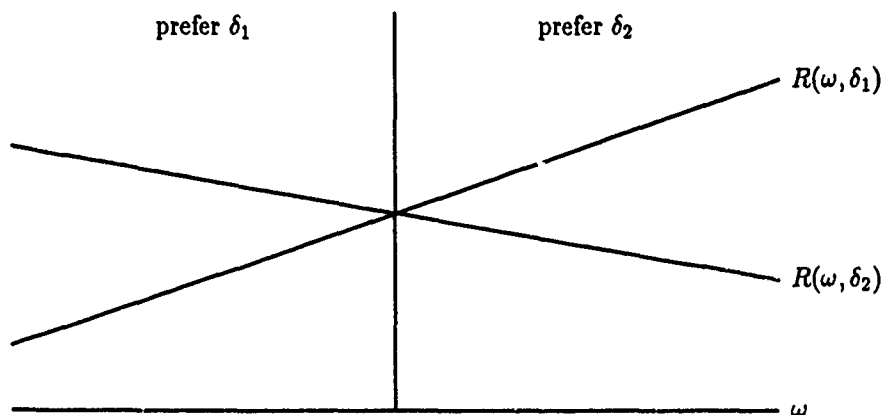


Figure 3: Incomplete comparison of decision rules through risk

but direct computation of a minimax rule from the definition is usually not possible. Instead, the Bayes and equalizer approaches provide an indirect strategy for finding minimax rules. A standard result from statistical decision theory states that a Bayes equalizer rule is minimax:

Theorem 1 Let π be a distribution on Ω , and suppose that the decision rule δ is Bayes against π . If δ is an equalizer rule, then δ is minimax.

Proof See [Ferguson, 1967, p. 90] or [McKendall, 1990, p. 271].

Thus, the strategy for finding a minimax rule is first to construct an equalizer rule and second to show that it is Bayes against some probability distribution on Ω . Theorem 2 gives an extension of this strategy:

Theorem 2 Let π be a distribution on Ω , and suppose that the decision rule δ is Bayes against π . Suppose that there is a constant C such that the following two conditions are met:

1. $R(\omega, \delta) \leq C$ for all $\omega \in \Omega$.
2. $P\{\omega : R(\omega, \delta) = C\} = 1$.

Then δ is minimax.

Proof See [Ferguson, 1967, p. 90] or [McKendall, 1990, p. 272].

The probability distribution of these theorems is a mathematical tool; it has no interpretation for application. It is a least-favorable distribution. A distribution π_0 on Ω is least favorable if

$$\inf_{\delta} E^{\pi_0}[R(\omega, \delta)] = \sup_{\pi} \inf_{\delta} E^{\pi}[R(\omega, \delta)].$$

(The superscripts indicates the distribution on Ω .)

Computation of a Bayes rule is usually easier than computation of a minimax rule from the definition. Theorem 3 outlines a strategy for finding a Bayes rule:

Theorem 3 Let π be a distribution on Ω , and let $\pi(\cdot|z)$ be the conditional distribution on Ω given the observation $Z = z$. If for all z ,

$$E^{\pi(\cdot|z)}[L(\omega, \delta(z))] = \inf_a E^{\pi(\cdot|z)}[L(\omega, a)],$$

then δ is Bayes against π .

Proof See [Ferguson, 1967, pp. 43–45].

The conditional distribution $\pi(\cdot|z)$ on Ω is the posterior distribution on Ω . The expected value under $\pi(\cdot|z)$ of the loss $L(\omega, a)$ is the posterior expected loss of an action a . Thus, a strategy for finding a Bayes rule against a prior distribution is to minimize the posterior expected loss under the corresponding posterior distribution.

Utility of Decision Theory

This decision-theoretic formulation of the location problem has several features. First, standard estimation and robust estimation coincide within the framework of statistical decision theory. The only difference is the specification of the parameter space: $\Omega = \Theta$ or $\Omega = \Theta \times \mathcal{F}$. The tools of statistical decision theory, however, apply to either specification. Second, decision theory incorporates prior information about the unknown parameters through the minimax criterion by optimizing over $\omega \in \Omega$. Third, a decision problem accounts for the consequences of the estimate through the loss function. Zero-one (e) loss, in particular, models error tolerance: An estimate within e of θ is sufficiently close and so incurs no penalty, and an estimate greater than e from θ is too far and thus incurs full penalty. Also, zero-one loss is independent of F . Finally, a minimax estimator $\delta^*(Z)$ based on zero-one (e) loss induces an optimal fixed-size ($2e$) confidence procedure that maximizes the confidence coefficient among all fixed-size ($2e$) confidence procedures. This fixed-size confidence procedure induced by an estimator δ of θ is

$$C_{\delta}(Z) := [\delta(Z) - e, \delta(Z) + e].$$

The confidence coefficient is $\inf_{\omega} P_{\omega}\{C_{\delta}(Z) \ni \theta\}$, where $P_{\omega}\{C_{\delta}(Z) \ni \theta\}$ is the probability under ω that the con-

fidence interval covers θ . If δ^* is a minimax rule, then

$$\inf_{\omega} P_{\omega}\{C_{\delta^*}(Z) \ni \theta\} = \sup_{\theta} \inf_{\omega} P_{\omega}\{C_{\delta}(Z) \ni \theta\}.$$

This confidence procedure provides a test of hypothesis that two measurements Z_1 and Z_2 are consistent.

5 Examples

Example This example gives a minimax rule for the location or mean θ of a measurement $Z \sim \mathcal{N}(\theta, 1)$ with $\theta \in \{-1, 0, 1\}$ when the error tolerance ϵ is 0.

The random variable Z has the structure $Z = \theta + V$ where $V \sim \mathcal{N}(0, 1)$. The possible values of θ are the elements of $\Theta = \{-1, 0, 1\}$. This example is a standard-estimation problem since the nominal distribution F is known. Thus $\Omega = \Theta$ or $\omega = \theta$. Also, the action space \mathcal{A} is Θ . The loss function is the zero-one (0) loss function:

$$L_0(\theta, a) := \begin{cases} 0 & \text{if } a = \theta \\ 1 & \text{if } a \neq \theta \end{cases}$$

The minimax decision rule δ^* is this:

$$\delta^*(z) = \begin{cases} -1 & \text{if } z \leq -0.803 \\ 0 & \text{if } -0.803 < z < 0.803 \\ 1 & \text{if } 0.803 \leq z \end{cases}$$

This rule implies, for example, that the estimate corresponding to the observation $Z = 0.5$ is $\hat{\theta} = 0$. Similarly, the estimate corresponding to any observation $Z \geq 0.803$ is $\hat{\theta} = 1$.

The risk function of δ^* is this:

$$\begin{aligned} R(-1, \delta^*) &= 1 - F(-0.803 + 1) \\ R(0, \delta^*) &= 2F(-0.803) \\ R(1, \delta^*) &= F(0.803 - 1) \end{aligned}$$

This decision rule is an equalizer rule with risk 0.422.

Furthermore, the rule δ^* is Bayes against the distribution on Θ that assigns these probabilities:

$$\begin{aligned} p(-1) &= 0.2982 \\ p(0) &= 0.4036 \\ p(1) &= 0.2982 \end{aligned}$$

(See [McKendall, 1990] for the analysis underlying this example and for similar problems in standard estimation.) \square

Example This example gives a minimax rule for the location θ of a measurement $Z \sim \mathcal{N}(\theta, 1)$ with $\theta \in [-0.3, 0.3]$ when the error tolerance ϵ is 0.1.

This example is also a standard-estimation problem. The parameter space and action space both are the interval $[-0.3, 0.3]$. The zero-one (0.1) loss function is this:

$$L_{0.1}(\theta, a) := \begin{cases} 0 & \text{if } |\theta - a| \leq 0.1 \\ 1 & \text{if } |\theta - a| > 0.1 \end{cases}$$

The minimax decision rule δ^* is this:

$$\delta^*(z) = \begin{cases} -\delta^*(-z) & \text{if } z < 0 \\ 0 & \text{if } 0 \leq z < a \\ z - a & \text{if } a \leq z < a + 0.2 \\ 0.2 & \text{if } a + 0.2 \leq z \end{cases} \quad (1)$$

Here $a = 0.3992$. (See figure 4.) This rule has $|\delta^*(z)| \leq 0.2$ since the error tolerance is 0.1.

The risk function of δ^* is this:

$$R(\theta, \delta^*) = \begin{cases} R(-\theta, \delta^*) & \text{if } \theta < 0 \\ 2F(-a - 0.1) & \text{if } 0 \leq \theta < 0.1 \\ F(-a - 0.1) & \text{if } \theta = 0.1 \\ F(a - 0.1) & \text{if } 0.1 < \theta \leq 0.3 \end{cases}$$

This decision rule has constant risk (0.6176) except for the points $\theta = \pm 0.1$, which have smaller risk. Since these points together have zero probability under any continuous distribution, this rule is essentially an equalizer rule. In particular, theorem 2 applies to this rule.

The rule δ^* is Bayes against the distribution on Θ that has this density function:

$$p(\theta) = \begin{cases} 1.62 & \text{if } -0.3 \leq \theta \leq -0.1 \\ 1.76 & \text{if } -0.1 < \theta < 0.1 \\ 1.62 & \text{if } 0.1 \leq \theta \leq 0.3 \end{cases}$$

(See [Zeytinoglu and Mintz, 1984] for the analysis underlying this example.) \square

Example This example gives a minimax rule for the location θ of a measurement $Z \sim \mathcal{N}(\theta, \sigma^2)$ with $\theta \in [-0.3, 0.3]$ and some $\sigma \leq 0.25$ when the error tolerance is 0.1.

This example is a robust-estimation problem since the scale and hence the nominal distribution $F \sim \mathcal{N}(0, \sigma^2)$ are uncertain. The uncertainty class is

$$\mathcal{F} = \{\mathcal{N}(0, \sigma^2), \sigma \leq 0.25\}.$$

The parameter space Ω is $\Theta \times \mathcal{F}$ or, equivalently, $[-0.3, 0.3] \times (0, 0.25]$. The action space and loss function are the same as those of the previous example.

This problem reduces to a standard-estimation problem since the largest possible scale is sufficiently small relative to the error tolerance. The minimax rule for this example is the minimax rule for the standard-estimation problem of the last example with the nominal distribution replaced by $\mathcal{N}(0, 0.25^2)$. In particular, the minimax rule is given by definition 1 with $a = 0.0808$.

(See [Zeytinoglu and Mintz, 1988] for the analysis underlying this example. See [Martin, 1987] and [McKendall, 1990] for other problems in robust estimation.) \square

References

- [Berger, 1985] J.O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, New York, second edition, 1985.
- [DeGroot, 1970] M.H. DeGroot. *Optimal Statistical Decisions*. McGraw-Hill Book Company, New York, 1970.
- [Ferguson, 1967] T.S. Ferguson. *Mathematical Statistics: A Decision Theoretic Approach*. Academic Press, Inc., Orlando, FL, 1967.
- [Kamberova and Mintz, 1990] G. Kamberova and M. Mintz. Robust multi-sensor fusion: a decision theoretic approach. In these Proceedings, October 1990.

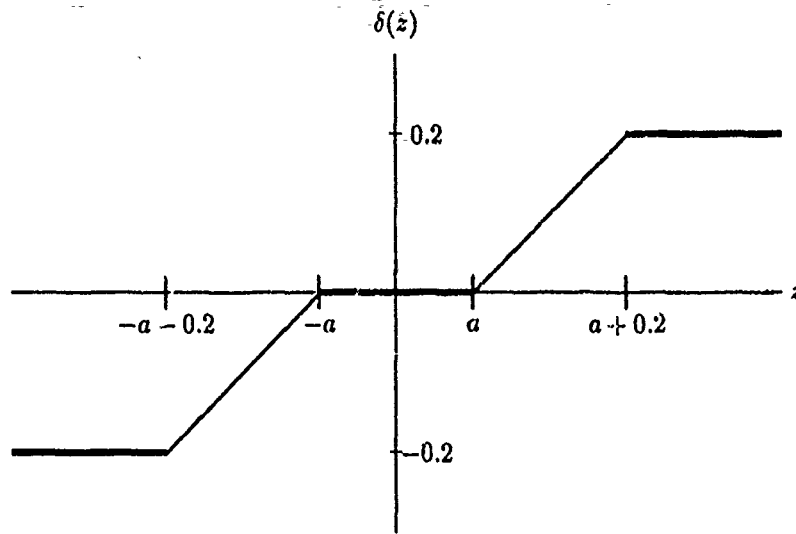


Figure 4: A minimax rule when $F \sim \mathcal{N}(0, 1)$, $\Theta = [-0.3, 0.3]$, and $e = 0.1$.

- [Martin, 1987] K.E. Martin. *Randomized Robust Confidence Procedures*. PhD thesis, Department of Systems Engineering, University of Pennsylvania, December 1987.
- [Martin and Mintz, 1984] K.E. Martin and M. Mintz. Randomized robust confidence procedures. In *Proceedings of the Twenty-Second Annual Allerton Conference on Communication, Control, and Computing*, pages 309–317, University of Illinois, October 1984.
- [McKendall, 1990] R. McKendall. *Minimax Estimation of a Discrete Location Parameter for a Continuous Distribution*. PhD dissertation. GRASP Lab technical report MIS-CS-90-28, Department of Computer and Information Science, University of Pennsylvania, May 1990.
- [McKendall and Mintz, 1990] R. McKendall and M. Mintz. Non-monotonic decision rules for sensor fusion. In these Proceedings, October 1990.
- [Zeytinoglu and Mintz, 1984] M. Zeytinoglu and M. Mintz. Optimal fixed sized confidence procedures for a restricted parameter space. *The Annals of Statistics*, 12(3):945–957, September 1984.
- [Zeytinoglu and Mintz, 1988] M. Zeytinoglu and M. Mintz. Robust optimal fixed sized confidence procedures for a restricted parameter space. *The Annals of Statistics*, 16(3):1241–1253, September 1988.

Robust Multi-Sensor Fusion: A Decision-Theoretic Approach

Gerda Kamberova and Max Mintz*

GRASP Laboratory

Department of Computer and Information Science

University of Pennsylvania

Philadelphia, PA 19104-6389

Abstract

Many tasks in active perception require that we be able to combine different information from a variety of sensors that relate to one or more features of the environment. Prior to combining these data, we must test our observations for consistency. The purpose of this paper is to examine sensor fusion problems for linear location data models using statistical decision theory (SDT). The contribution of this paper is the application of SDT to obtain: (i) a robust test of the hypothesis that data from different sensors are consistent; and (ii) a robust procedure for combining the data that pass this preliminary consistency test. Here, robustness refers to the statistical effectiveness of the decision rules when the probability distributions of the observation noise and the a priori position information associated with the individual sensors are uncertain. The standard linear location data model refers to observations of the form: $Z = \theta + V$, where V represents additive sensor noise and θ denotes the "sensed" parameter of interest to the observer. While the theory addressed in this paper applies to many uncertainty classes, the primary focus of this paper is on asymmetric and/or multimodal models, that allow one to account for very general deviations from nominal sampling distributions. This paper extends earlier results in SDT and multi-sensor fusion obtained by [Zeytinoglu and Mintz, 1984], [Zeytinoglu and Mintz, 1988], and [McKendall and Mintz, 1988].

1 Introduction

Our research in active sensing is based on the theory and application of multiple sensors in the exploration of environments that are characterized by significant a priori uncertainties. In addition to uncertainty in the environment, the sensors themselves exhibit noisy behavior. While good engineering practice can reduce certain noise

components, it is impractical if not impossible to eliminate them completely. Thus, all sensor measurements are uncertain. However, sensor errors can be modeled statistically, using both physical theory and empirical data. In developing these models, one recognizes that a single distribution is usually an inadequate description of sensor noise behavior. It is much more realistic and much safer to identify an envelope or class of distributions, one of whose members could represent the actual statistical behavior of the given sensor. This use of an uncertainty class (or equivalently: an envelope, set, or neighborhood) in distribution space, protects the system designer against the inevitable unpredictable changes that occur in sensor behavior. Reasons for uncertainty in statistical sensor models include: sporadic interference, drift due to aging, temperature variations, miscalibration, quantization, and other significant nonlinearities over the dynamic range of the sensor. The purpose of this paper is to examine a sensor fusion problem for linear location data models using statistical decision theory (SDT). The contribution of this paper is the application of SDT to obtain: (i) a robust test of the hypothesis that data from different sensors are consistent; and (ii) a robust procedure for combining the data that pass this preliminary consistency test. Here, robustness refers to the statistical effectiveness of the decision rules when the probability distributions of the observation noise and the a priori position information associated with the individual sensors are uncertain. The standard linear location data model refers to observations of the form: $Z = \theta + V$, where V represents additive sensor noise and θ denotes the "sensed" parameter of interest to the observer. The parameter θ is called a location parameter, since the distribution of Z is obtained from the distribution of V by a translation. While the location parameter fusion problem is only one of many possible fusion paradigms, it does provide a useful starting point for considering more complicated problems, e.g., nonlinear location sensor models of the form: $Z = h(\theta) + V$, where h denotes a given (nonlinear) function. It also provides a useful starting point for considering important generalizations of the location sensor model such as: $Z = h(\theta + V)$.

While the theory addressed in this paper applies to many uncertainty classes, the primary focus of this paper is on asymmetric and/or multimodal models, that allow one to account for very general deviations from

*Acknowledgement: Navy Contract N0014-88-K-0630; AFOSR Grants 88-0244, 88-0296; Army/DAAL 03-89-C-0031PRI, NSF Grants CISE/CDA 88-22719, IRI 89-06770, and the Dupont Corporation.

nominal sampling distributions. This paper extends earlier results in SDT and multi-sensor fusion obtained by [Zeytinoglu and Mintz, 1984], [Zeytinoglu and Mintz, 1988], and [McKendall and Mintz, 1988].

In the sequel we: (i) delineate several paradigms for robust fusion of multi-sensor linear location data; (ii) introduce some essential nomenclature and definitions from SDT; (iii) state the decision-theoretic results that this paper is based on; and (iv) present and discuss a methodology for robust fusion of multi-sensor linear location data.

Our presentation emphasizes the statement and application of the relevant theory. Proofs of theorems are omitted. The reader is referred to journal articles and reports for these details.

2 Paradigms for Sensor Fusion of Location Data

In this section we delineate several paradigms for robust fusion of location data. We restrict our attention to observations of one-dimensional location parameters. The results of this one-dimensional analysis can be applied to the multi-dimensional case by doing a component by component analysis. Alternatively, one can pursue a formal multi-dimensional extension of the methodology presented in this paper. This extension is part of our current research in sensor fusion.

The general one-dimensional paradigm is delineated as follows. We assume that we are given the sampled outputs of r sensor systems $\{S_i : 1 \leq i \leq r\}$. We denote the k^{th} sampled output of S_i , $1 \leq k \leq N_i$ by:

$$Z_{ik} = \mu_i + W_i + \theta_i + V_{ik}, \quad (2.1)$$

where:

- $a_i \leq \theta_i \leq b_i$, denotes an unknown location parameter with known bounds a_i and b_i . [The bounds a_i and b_i may assume infinite values.] In many applications there is a common interval of location parameter uncertainty for all sensors. However, there is no need to make this assumption in the following mathematical developments.
- μ_i , denotes a known constant (offset) associated with the position of sensor S_i with respect to a common origin.
- V_{ik} , denotes the additive observation noise associated with the k^{th} observation (sample) from S_i . The random variables $\{V_{ik} : 1 \leq k \leq N_i\}$ are assumed to be independent and identically distributed (i.i.d.). We further assume that the noise process associated with S_i is independent of the noise process associated with S_j , when $i \neq j$. Finally, we assume that the probability distribution of V_{ik} belongs to a given uncertainty class of distributions, \mathcal{F}_i . We do *not* assume that the noise processes associated with different sensors are identically distributed.
- W_i , denotes the uncertainty in the position of sensor S_i with respect to a common origin. We consider two cases: (i) the position uncertainty of S_i can be expressed by a known interval $[l_i, u_i]$ — with

no a priori probabilistic description; or (ii) the position uncertainty of S_i can be expressed by an unknown probability distribution from a given uncertainty class \mathcal{P}_i . In each case, we assume that the position uncertainty of S_i is independent of the observation noise $\{V_{ik} : 1 \leq k \leq N_i\}$, and independent of the observation noise and position uncertainty of the other sensors.

Remark 2.1 Without loss of generality, we can assume that the known offsets $\{\mu_i : 1 \leq i \leq r\}$ are each zero, since nonzero values can be subtracted from the observations $\{Z_{ik} : 1 \leq k \leq N_i\}$. Further, if the known, generally asymmetric, interval of uncertainty $[a_i, b_i]$ in θ_i is finite, then the observations $\{Z_{ik} : 1 \leq k \leq N_i\}$ can be shifted and the interval of uncertainty $[a_i, b_i]$ can be replaced by $[-d_i, d_i]$, where $d_i = (b_i - a_i)/2$. Similarly, we can assume the interval of sensor position uncertainty (where applicable) is also symmetric. Thus, (2.1) can be replaced by:

$$Z_{ik} = W_i + \theta_i + V_{ik}, \quad (2.2)$$

where: $1 \leq k \leq N_i$, $|\theta_i| \leq d_i$, and (where applicable) $|W_i| \leq \eta_i$, $1 \leq i \leq r$.

The uncertainty classes \mathcal{F}_i and (where applicable) \mathcal{P}_i , $1 \leq i \leq r$, denote subsets in the space of probability distributions that are deemed to characterize the uncertainty in the specifications of the sampling distributions. Models for several uncertainty classes are described in Sections 4 and 6.

As stated in the introduction, the purpose of this paper is to examine a sensor fusion problem for location information using SDT. The contribution of this paper is the application of SDT to obtain: (i) a robust test of the hypothesis that data from different sensors are consistent, i.e., testing the hypothesis that $\theta_i = \theta_j$, $1 \leq i < j \leq r$; and (ii) a robust procedure for combining the data that pass this preliminary consistency test. Again, robustness refers to the statistical effectiveness of the decision rules when the probability distributions of the observation noise and the a priori position information of the individual sensors are uncertain.

In the following section, we introduce the notions of robust minimax decision rules and robust confidence procedures. These concepts provide the basis for the developments in the remainder of this paper.

3 Nomenclature and Definitions from SDT

The standard statement of a minimax location parameter estimation problem includes as given: a parameter space Ω ; a space of actions \mathcal{A} ; a loss function L defined on $\mathcal{A} \times \Omega$; and a CDF F . If the underlying CDF is imprecisely known, then this standard minimax decision model must be reformulated to account for this additional uncertainty. Statistical decision rules that are applicable in this more general problem setting are called robust procedures.

This paper considers robust fixed size confidence procedures for a restricted parameter space. These robust confidence procedures are based, in turn, on the solution of a related robust minimax decision problem:

Basic Minimax Decision Problem (MDP): Let Z denote a vector of N i.i.d. observations of a scalar random variable with CDF $F(z - \theta)$, where $F \in \mathcal{F}$, a given uncertainty class. Let $\Omega = \mathcal{A} = [-d, d]$, and define a zero-one loss function L on $\mathcal{A} \times \Omega$:

$$L(a, \theta) = \begin{cases} 0, & |a - \theta| \leq e; \\ 1, & |a - \theta| > e; \end{cases} \quad (3.1)$$

where $e > 0$, is given. Further, let $R(\delta, \theta, F) = E[L(\delta, \theta) | \theta, F]$ denote the risk function of the decision rule δ given $\theta \in \Omega$ and $F \in \mathcal{F}$.

Definition 3.1 An estimator δ^* is said to be a robust minimax estimator for θ , if for all δ :

$$\sup_{\substack{\theta \in \Omega \\ F \in \mathcal{F}}} R(\delta^*, \theta, F) \leq \sup_{\substack{\theta \in \Omega \\ F \in \mathcal{F}}} R(\delta, \theta, F).$$

Based on these definitions and assumptions, we seek a robust minimax estimator δ^* for θ . For brevity, we restrict our consideration to the case when d/e is an integer ≥ 2 .

Observation 3.1 The connection between the robust minimax rule $\delta^*(Z)$ and a robust fixed size confidence procedure is obtained by noting that:

$$C^*(Z) = [\delta^*(Z) - e, \delta^*(Z) + e]$$

can be interpreted as a robust confidence procedure of size $2e$ that has the highest confidence coefficient $\inf_{\theta, F} P_{\theta, F}[\theta \in C^*(Z)]$.

Sections 4, 5, and 6 of this paper are organized as follows:

Section 4 presents solutions of two related single-sample minimax estimation problems where F is given. These results provide the basis for the solutions to the robust minimax estimation problems where $F \in \mathcal{F}$.

Section 5 extends the results of Section 4 to the multi-sample case.

Section 6 develops a theory and methodology for robust sensor fusion of location information based on the theory presented in Sections 4 and 5.

4 Minimax and Robust Minimax Rules

Throughout Section 4 we consider the single-sample decision problem MDP ($N = 1$).

4.1 Minimax Rules

Minimax problems are special cases of robust minimax problems in the sense that \mathcal{F} contains a single CDF F . We begin with two minimax estimation problems that are defined by the zero-one loss function L (3.1). The solutions to these single-sample estimation problems provide the basis for solutions to both the single-sample and multi-sample robust minimax estimation problems. These preliminary results require Definitions 4.1-4.2 and are summarized by Theorems 4.1-4.2.

Definition 4.1 Let \mathcal{C}_a denote the class of nonrandomized, monotone nondecreasing decision rules $\delta: E^1 \rightarrow \mathcal{A}$, where: $\mathcal{A} = [-d, d]$. Let $\Delta_a \subset \mathcal{C}_a$ denote the set of rules $\delta(t)$, defined for $t \in (-\infty, \infty)$ by (4.1), where: $i = 1, 2, \dots, n$ and $-\infty < a_{-n} \leq \dots \leq a_{-2} \leq a_{-1} \leq a_0 \leq a_1 \leq a_2 \leq \dots \leq a_n < \infty$, $d = (2n + 1)e + c$, and c equals zero (e) if d is an odd (even) multiple of e . Note that the parameter a_0 is relevant only when c equals e .

Observation 4.1 Let L denote the zero-one loss function (3.1). If the CDF F is continuous, then for each $\delta \in \Delta_a$, the risk function $R(\delta, \theta, F)$ is (4.2), where $i = 1, 2, \dots, n - 1$. For each $\delta \in \Delta_a$, $R(\delta, \theta, F)$ is a piecewise constant function of θ over the sets of a finite partition of Ω , and the maximum of $R(\delta, \theta, F)$ occurs at one or more of the nondegenerate intervals. The risk expression (4.2) can be readily modified to include CDF's F that are discontinuous. The generalized risk function $R(\delta, \theta, F)$ is again a piecewise constant function of θ over the sets of a finite partition of Ω expressed in (4.2).

Theorem 4.1 Let L denote the zero-one loss function (3.1), and T be a scalar random variable with given CDF $F(t - \theta)$. If F is absolutely continuous with respect to Lebesgue measure, has convex support, and possesses a (strictly) monotone likelihood ratio, then there exists a globally minimax (admissible) Bayes rule $\delta^* \in \Delta_a$ and a least favorable prior distribution λ^* .

Proof: See [Kamberova and Mintz, 1990].

Remark 4.1 $R(\delta^*, \theta, F)$ and λ^* have the following characteristics:

- The minimax rule δ^* is an "almost" equalizer rule, in the sense that the nondegenerate piecewise constant segments of the risk function are equalized to the minimax risk by a suitable choice of the parameter vector $a = (a_{-n}, \dots, a_n)^T$.
- The least favorable prior distribution λ^* is defined by a density function that is piecewise constant.

Remark 4.2 Theorem 4.1 extends the basic minimax results of [Zeytinoglu and Mintz, 1984] by allowing the inclusion of CDF's F that are asymmetric.

Definition 4.2 A rule is (robust) \mathcal{D} -minimax if it is (robust) minimax within the class \mathcal{D} . A rule is \mathcal{D} -Bayes if it is Bayes within the class \mathcal{D} . A rule is \mathcal{D} -admissible if it is admissible within the class \mathcal{D} .

In the following theorem we weaken the hypothesis of Theorem 4.1 by dropping the monotone likelihood ratio condition, and obtain a \mathcal{C}_a -minimax result.

Theorem 4.2 Let L denote the zero-one loss function (3.1), and T be a scalar random variable with given CDF $F(t - \theta)$. If F is absolutely continuous with respect to Lebesgue measure and has convex support, then there exists a \mathcal{C}_a -minimax rule $\delta^* \in \Delta_a$.

Proof: See [Kamberova and Mintz, 1990].

Remark 4.3 $R(\delta^*, \theta, F)$ has the following characteristic:

- The \mathcal{C}_a -minimax rule δ^* is an "almost" equalizer rule in the sense of Remark 4.1.

Remark 4.4 Theorem 4.2 extends the basic \mathcal{C} -minimax results of [Zeytinoglu and Mintz, 1984] by allowing the inclusion of CDF's F that are asymmetric and/or multimodal.

4.2 Robust Minimax Rules

In this section we define two uncertainty classes \mathcal{F} , and delineate the solutions to the corresponding robust minimax and robust \mathcal{C}_a -minimax estimation problems. These results require Definitions 4.3-4.4 and are summarized by Theorems 4.3-4.6.

$$\delta(t) = \begin{cases} d-e, & c+a_n+2ne \leq t; \\ t-a_i, & c+a_i+2(i-1)e \leq t < c+a_i+2ie; \\ 2(i-1)e+c, & c+2(i-1)e+a_{i-1} \leq t < c+a_i+2(i-1)e; \\ t-a_0, & -c+a_0 < t < c+a_0; \\ -2(i-1)e-c, & -c+a_{-i}-2(i-1)e < t \leq -c+a_{-i+1}-2(i-1)e; \\ t-a_{-i}, & -c+a_{-i}-2ie < t \leq -c+a_{-i}-2(i-1)e; \\ -d+e, & t \leq -c+a_{-n}-2ne; \end{cases} \quad (4.1)$$

$$R(\delta, \theta, F) = \begin{cases} F(a_n-e), & d-2e < \theta \leq d; \\ F(a_{n-1}-e), & \theta = d-2e; \\ F(a_i-e)+1-F(a_{i+1}+e), & c+(2i-1)e < \theta \leq c+(2i+1)e; \\ F(a_{-1+c/e}-e)+1-F(a_1+e), & \theta = c+e; \\ F(a_0-e)+1-F(a_1+e), & -c+e < \theta < c+e; \\ F(a_{-1}-e)+1-F(a_1+e), & \theta = -c+e; \\ F(a_{-1}-e)+1-F(a_1+e), & c-e < \theta < -c+e; \\ F(a_{-2+c/e}-e)+1-F(a_{-1+c/e}+e), & \theta = c-e; \\ F(a_{-1}-e)+1-F(a_0+e), & -c-e < \theta < c-e; \\ F(a_{-2}-e)+1-F(a_{-1}+e), & \theta = -c-e; \\ F(a_{-(i+1)}-e)+1-F(a_{-i}+e), & -c-(2i+1)e \leq \theta < -c-(2i-1)e; \\ 1-F(a_{-n+1}+e), & \theta = -d+2e; \\ 1-F(a_{-n}+e), & -d \leq \theta < -d+2e; \end{cases} \quad (4.2)$$

Definition 4.3 Let \mathcal{F} denote an uncertainty class with upper-envelope F_u :

$$\mathcal{F} = \{F: F(x^-) \leq F_u(x), x \leq s; F(x) \geq F_u(x), x > s\}, \quad (4.3)$$

where F_u is absolutely continuous with respect to Lebesgue measure and has convex support.

Remark 4.5 The CDF F_u defines the upper-envelope of \mathcal{F} (4.3) in the sense that: $F(x) \leq F_u(x)$ for all $F \in \mathcal{F}$, and $x < s$. The upper-envelope CDF F_u is permitted to be substochastic, i.e., F_u can have less than unit probability mass. Thus, all ϵ -contamination models can be represented by a simple generalization of \mathcal{F} (4.3).

The following theorem extends the results of Theorem 4.1 to the single-sample robust minimax estimation problem.

Theorem 4.3 Let \mathcal{F} denote the uncertainty class (4.3) with upper-envelope F_u . Assume F_u possesses a (strictly) monotone likelihood ratio. Let δ^* denote the minimax rule obtained through Theorem 4.1 based on CDF F_u . There exists a bound $B(d/e, F_u)$, such that if $e \geq B$, then δ^* is a robust minimax (admissible) Bayes rule.

Proof: See [Kamberova and Mintz, 1990].

The following theorem extends the results of Theorem 4.2 to the single-sample robust \mathcal{C}_a -minimax estimation problem.

Theorem 4.4 Let \mathcal{F} denote the uncertainty class (4.3) with upper-envelope F_u . Let δ^* denote the \mathcal{C}_a -minimax rule obtained through Theorem 4.2 based on CDF F_u . There exists a bound $B(d/e, F_u)$, such that if $e \geq B$, then δ^* is a robust \mathcal{C}_a -minimax rule.

Proof: See [Kamberova and Mintz, 1990].

Definition 4.4 Let \mathcal{F} denote the uncertainty class:

$$\mathcal{F} = \{F(\cdot) = F_0((\cdot - \tau)/\sigma): |\tau| \leq \eta; \sigma \leq \sigma_u\}, \quad (4.4)$$

where: $\eta > 0$ and $\sigma_u > 0$ denote given bounds, and F_0 denotes a given CDF that is symmetric about zero, and absolutely continuous with respect to Lebesgue measure.

Remark 4.6 The uncertainty class \mathcal{F} (4.4) models underlying uncertainty in both location and scale for a symmetric distribution F_0 . Without loss of generality, we can assume $\sigma_u = 1$.

Remark 4.7 The delineation of robust minimax rules and robust \mathcal{C}_a -minimax rules for the estimation problem defined by the zero-one loss function L (3.1), and the uncertainty class \mathcal{F} (4.4) is obtained by determining the joint worst-case behavior of the parameters: θ , τ , and σ . By worst case, we mean those combinations of parameter values that lead to maximum risk. In carrying out this worst-case analysis, it is necessary to consider two cases: d/e is odd, and d/e is even. For brevity, we restrict our analysis to the even case. The complete analysis appears in [Kamberova and Mintz, 1990].

Observation 4.2 Let $d = (2n+2)e$, $n \geq 0$. There exist bounds $B_1(d/e, \sigma_u, F_0)$ and $B_2(d/e, \sigma_u, F_0)$ such that if $\eta \leq B_1$ and $e \geq B_2$, then the joint worst-case behavior of θ , τ , and σ is: $\tau = -\eta$ when $\theta > 0$; $\tau = \eta$ when $\theta < 0$; and $\sigma = \sigma_u$ for all θ .

Observation 4.3 As a consequence of the underlying even and odd symmetry in this decision problem, which is reflected by the worst-case analysis, we can restrict our attention to rules $\delta \in \Delta_a$ that possess odd symmetry about zero ($a_0 = 0$ and $a_{-i} = -a_i$). We denote this subset of Δ_a by Δ .

Observation 4.4 If the relation between the parameters θ , τ , and σ is defined by the worst-case analysis of Observation 4.2, then for any $\delta \in \Delta$, the worst-case risk (for $\theta > 0$) is (4.5), where: $\sigma_u = 1$, and $d = (2n+2)e$, $n \geq 0$. We can restrict our attention to the domain $\theta > 0$ due to the even and odd symmetry in this decision problem.

$$R(\delta, \theta, F_0) = \begin{cases} F_0(a_n + \eta - e), & d - 2e < \theta \leq d; \\ F_0(a_{n-1} + \eta - e), & \theta = d - 2e; \\ F_0(-a_n - \eta - e) + F_0(a_{n-1} + \eta - e), & d - 4e < \theta < d - 2e; \\ \cdot & \cdot \\ \cdot & \cdot \\ F_0(-a_2 - \eta - e) + F_0(a_1 + \eta - e), & 2e < \theta < 4e; \\ F_0(-a_2 - \eta - e) + F_0(\eta - e), & \theta = 2e; \\ F_0(-a_1 - \eta - e) + F_0(\eta - e), & 0 < \theta < 2e; \end{cases} \quad (4.5)$$

Lemma 4.1 If F_0 is absolutely continuous with respect to Lebesgue measure and has convex support, then there exists a choice of parameters $\{a_i : 1 \leq i \leq n\}$ that equalize the nondegenerate piecewise constant segments of the risk function (4.5). The corresponding rule δ^* is an "almost" equalizer rule.

The following theorem delineates the existence and structure for single-sample robust minimax rules in the case of the joint location-scale uncertainty class \mathcal{F} (4.4).

Theorem 4.5 Let \mathcal{F} denote the location-scale uncertainty class (4.4) based on the symmetric CDF F_0 . Assume F_0 possesses a (strictly) monotone likelihood ratio and has convex support. Let δ^* denote the rule obtained through Lemma 4.1. There exists bounds $B_1(d/e, \sigma_u, F_0)$, and $B_2(d/e, \sigma_u, F_0)$ such that if $\eta \leq B_1$, and $e \geq B_2$, then δ^* is a robust minimax (admissible) Bayes rule.

Proof: See [Kamberova and Mintz, 1990].

In the following theorem we weaken the hypothesis of Theorem 4.5 by dropping the monotone likelihood ratio condition, and obtain a robust C_a -minimax result.

Theorem 4.6 Let \mathcal{F} denote the location-scale uncertainty class (4.4) based on the CDF F_0 . Assume F_0 has convex support. Let δ^* denote the rule obtained through Lemma 4.1. There exists a bound $B(d/e, \sigma_u, F_0)$ such that if $e \geq B$, then δ^* is a robust C_a -minimax rule.

Proof: See [Kamberova and Mintz, 1990].

5 The Multi-Sample Case

This section extends the robust minimax results of Theorems 4.3-4.6 to the multi-sample problem ($N > 1$) by restricting the class of estimators to rules of the form $\delta(T(\mathbf{Z}))$, where: $\delta \in C_a$, T is a real-valued function of \mathbf{Z} , and $T(\mathbf{Z})$ possesses a CDF that depends on θ as a location parameter, is absolutely continuous with respect to Lebesgue measure, and has convex support. Examples of candidate T statistics include: the sample mean, the sample median, and other linear combinations of order statistics. In the remainder of this section we consider the sample median.

Definition 5.1 Let Z_M denote the median of the N observations \mathbf{Z} . [If N is even, $Z_M = (Z_{[N/2]} + Z_{[(N/2)+1]})/2$.] The decision rule $\delta^*(Z_M)$, defined by the composition $\delta^* \circ Z_M$, is said to be a median-minimax estimator for θ , if δ^* is a minimax rule in the usual sense. The respective definitions of robust median-minimax rules, C_a -median-minimax rules, and robust C_a -median-minimax rules are obtained as before.

The median statistic $T(\mathbf{Z}) = Z_M$ possesses several properties that are used in obtaining Theorems 5.1-5.4. These properties are stated in Observations 5.1-5.2.

Observation 5.1 The centered median statistic $Z_M - \theta$ preserves the upper-envelope of the uncertainty class \mathcal{F} (4.3). Further, the CDF of $Z_M - \theta$ preserves absolute continuity with respect to Lebesgue measure and convex support.

Observation 5.2 The median statistic Z_M preserves location ordering for fixed scale, and scale ordering for fixed location in the uncertainty class \mathcal{F} (4.4). Further, the CDF of Z_M preserves absolute continuity with respect to Lebesgue measure and convex support.

The following theorem extends the results of Theorem 4.3 to the multi-sample robust minimax estimation problem.

Theorem 5.1 Let $N > 1$ and \mathcal{F} denote the uncertainty class (4.3) with upper-envelope F_u . Let F_{uM} denote the CDF of the centered sample median $Z_M - \theta$, where the underlying common CDF is F_u . Assume F_{uM} possesses a (strictly) monotone likelihood ratio. Let δ^* denote the minimax rule obtained through Theorem 4.1 based on CDF F_{uM} . There exists a bound $B(d/e, N, F_u)$, such that if $e \geq B$, then δ^* is a robust median-minimax (median-admissible) median-Bayes rule.

Proof: See [Kamberova and Mintz, 1990].

The following theorem extends the results of Theorem 4.4 to the multi-sample robust C_a -minimax estimation problem.

Theorem 5.2 Let $N > 1$ and \mathcal{F} denote the uncertainty class (4.3) with upper-envelope F_u . Let F_{uM} denote the CDF of the centered sample median $Z_M - \theta$, where the underlying common CDF is F_u . Let δ^* denote the C_a -minimax rule obtained through Theorem 4.2 based on CDF F_{uM} . There exists a bound $B(d/e, N, F_u)$, such that if $e \geq B$, then δ^* is a robust C_a -median-minimax rule.

Proof: See [Kamberova and Mintz, 1990].

The following theorem extends the results of Theorem 4.5 to the multi-sample robust minimax estimation problem.

Theorem 5.3 Let $N > 1$ and \mathcal{F} denote the location-scale uncertainty class (4.4) based on the symmetric CDF F_0 . Assume F_0 has convex support. Let F_{0M} denote the CDF of the sample median, where the underlying common CDF is F_0 . Assume F_{0M} possesses a (strictly) monotone likelihood ratio. Let δ^* denote

the rule obtained through Lemma 4.1 based on the CDF F_{0M} . There exists bounds $B_1(d/e, N, \sigma_u, F_0)$, and $B_2(d/e, N, \sigma_u, F_0)$ such that if $\eta \leq B_1$, and $e \geq B_2$, then δ^* is a robust median-minimax (median admissible) median-Bayes rule.

Proof: See [Kamberova and Mintz, 1990].

The following theorem extends the results of Theorem 4.6 to the multi-sample robust C_a -minimax estimation problem.

Theorem 5.4 Let $N > 1$ and \mathcal{F} denote the location-scale uncertainty class (4.4) based on the symmetric CDF F_0 . Assume F_0 has convex support. Let F_{0M} denote the CDF of the sample median, where the underlying common CDF is F_0 . Let δ^* denote the rule obtained through Lemma 4.1 based on the CDF F_{0M} . There exists a bound $B(d/e, N, \sigma_u, F_0)$ such that if $e \geq B$, then δ^* is a robust C_a -median-minimax rule.

Proof: See [Kamberova and Mintz, 1990].

6 Robust Fusion of Location Information

6.1 Preliminary Remarks

In this section we develop a theory and methodology for robust fusion of multi-sensor location information based on Sections 4 and 5. Our approach contains two distinct phases:

- **Phase I** provides a test of the hypothesis $\theta_i = \theta_j$, that the location data (2.2) from sensor \mathcal{S}_i are consistent with the location data from sensor \mathcal{S}_j , where $i < j$.
- **Phase II** provides a means of combining the location data from the individual data sets that "pass" the Phase I test, i.e., those deemed to be consistent.

In both phases of this process, we seek procedures that are robust to heavy-tailed deviations from the nominal sampling distribution, such as exhibited in ϵ -contamination uncertainty classes. Our usage of "robust" is also intended to imply that the procedures have satisfactory behavior when the actual sampling distribution coincides with the nominal, e.g., a given Gaussian distribution.

6.2 Sample Sizes and Uncertainty Classes

In developing suitable consistency tests, there are three domains of sample sizes to address: (i) the single sample case, $N = 1$; (ii) the small sample case, $1 < N \leq 20$; and (iii) the large sample case, $N > 20$. In defining these classes, it is important to observe that the transition ($N = 20$) between the small sample and large sample cases is not a precise threshold value — the appropriate selection of this threshold is dependent on the uncertainty classes that define the given decision problem. The sample size for each sensor \mathcal{S}_i is denoted by N_i , $1 \leq i \leq r$. The sample sizes N_i and N_j can belong to different sample size domains.

The selection of appropriate sensor noise uncertainty classes $\{\mathcal{F}_i : 1 \leq i \leq r\}$ is an important issue in the development of a methodology for robust fusion of

multi-sensor location information. Since, at the minimum, we seek to account for the occurrence of noise distributions with heavy tails, it is appropriate to consider both ϵ -contamination uncertainty classes as well as joint location-scale uncertainty classes. We consider two cases:

Case 1: We adopt an ϵ -contamination model \mathcal{F}_ϵ , for each sensor \mathcal{S}_i , $1 \leq i \leq r$; in particular, the ϵ_i -contaminated non-Gaussian model for sensor \mathcal{S}_i that is defined by:

$$\mathcal{F}_\epsilon = \{F : F = (1 - \epsilon_i)\Psi + \epsilon_i H\}, \quad (6.1)$$

where: (i) Ψ denotes a given asymmetric, (possibly) multi-modal CDF that is absolutely continuous with respect to Lebesgue measure, and has convex support, and (ii) the CDF H is arbitrary, and $0 < \epsilon_i < 1$. This uncertainty class is a simple generalization of the uncertainty class (4.3).

Case 2: We adopt a joint location-scale uncertainty class for each sensor \mathcal{S}_i , $1 \leq i \leq r$; in particular, the joint location-scale uncertainty class defined by (4.4), where F_0 is the $N(0, 1)$ CDF, and the location-scale bounds are η_i and σ_{u_i} .

6.3 Phase I — Robust Consistency Tests

Analysis of Case 1: The following procedure provides a robust test of the hypothesis that $\theta_i = \theta_j$, $i < j$.

Let \mathcal{M}_i denote the class of CDF's defined by the centered sample median Z_{M_i} of N_i i.i.d. samples with CDF $F \in \mathcal{F}_\epsilon$, (6.1), $1 \leq i \leq r$. Let \mathcal{M}_{ij} denote the class of CDF's defined by the difference of the centered sample medians $(Z_{M_i} - \theta_i) - (Z_{M_j} - \theta_j)$, where the CDF's of the centered sample medians $(Z_{M_i} - \theta_i)$ and $(Z_{M_j} - \theta_j)$ belong, respectively, to \mathcal{M}_i and \mathcal{M}_j , $1 \leq i < j \leq r$. It follows from these definitions that the class \mathcal{M}_{ij} is a set of distributions of the form (4.3). Further,

$$Z_{M_i} - Z_{M_j} = \theta_i - \theta_j + \nu_{ij}, \quad (6.2)$$

where: the CDF of ν_{ij} belongs to \mathcal{M}_{ij} ; and the a priori uncertainty in $\theta_i - \theta_j$ is given by the interval $[-d_{ij}, d_{ij}]$, where $d_{ij} = d_i + d_j$.

Hence, we can construct a robust fixed size ($2e$) confidence procedure for $\theta_i - \theta_j$. The parameter e is selected by the decision maker: (i) it defines the decision maker's tolerance to small errors between θ_i and θ_j ; and (ii) it is used to select the size of the statistical test. The desired procedure $[\delta^* - e, \delta^* + e]$ is obtained via Theorem 5.2. Finally, the test of the hypothesis $\theta_i = \theta_j$ is obtained as follows: we reject $\theta_i = \theta_j$ if $0 \notin [\delta^* - e, \delta^* + e]$. From this test we also obtain the minimum probability that $\theta_i - \theta_j \in [\delta^* - e, \delta^* + e]$. Examples of applications of this class of robust consistency tests appears in [Kamberova et al., 1990].

Analysis of Case 2: We follow the basic approach described in the analysis of case 1, but we replace the sample median statistics by the sample means. Here, the sample mean is useful, since the underlying uncertainty classes contain only Gaussian distributions. The robust consistency test is obtained via Theorem 5.3. The details appear in [Kamberova et al., 1990].

6.4 Phase II — Robust Fusion of Consistent Multi-Sensor Location Information

The following procedure provides a robust estimate of the common location parameter θ of r sensor data sets, $r \geq 3$. We observe at the outset that, when V_1 and V_2 possess very heavy tails, in general, it is not useful to attempt to combine two observations of the form:

$$Z_1 = \theta + V_1$$

$$Z_2 = \theta + V_2$$

by convex combination. For example, if V_1 and V_2 are independent Cauchy $C(0, 1)$ random variables, then any convex combination of Z_1 and Z_2 will be a $C(\theta, 1)$ random variable. Further, there are random variables with continuous unimodal symmetric density functions whose sample mean, for any sample size $N > 1$, has greater variability than any of its N i.i.d. components.

Analysis of Case 1: Let $\{Z_{M_i} : 1 \leq i \leq r\}$ denote the sample medians of r consistent data sets with common location parameter θ . To simplify the exposition, we further assume that the r sample medians are identically distributed. Let Z_{M_A} denote the median of the $\{Z_{M_i} : 1 \leq i \leq r\}$. Let \mathcal{M}_A denote the uncertainty class of the centered sample median $Z_{M_A} - \theta$. The uncertainty class \mathcal{M}_A is of the form (4.3). Thus, we can apply Theorem 5.2 to obtain a robust fixed size confidence procedure $[\delta^* - e, \delta^* + e]$ for θ . Examples of applications of this class of confidence procedures for the robust fusion of consistent multi-sensor location information appears in [Kamberova *et al.*, 1990].

Analysis of Case 2: We follow the basic approach described in the analysis of case 1, but we replace the sample median statistics by the sample means. Here, the sample mean is useful, since the underlying uncertainty classes contain only Gaussian distributions. A robust estimate of location is obtained via Theorem 5.3. The details appear in [Kamberova *et al.*, 1990].

References

- [Kamberova and Mintz, 1990] Gerda Kamberova and Max Mintz. *Decision Theory for Robust Multi-Sensor Fusion*. Technical Report GRASP Lab, University of Pennsylvania, Philadelphia, PA, 1990.
- [Kamberova *et al.*, 1990] Gerda Kamberova, Raymond McKendall, and Max Mintz. *Robust Multi-Sensor Fusion*. Technical Report GRASP Lab, University of Pennsylvania, Philadelphia, PA, 1990.
- [McKendall and Mintz, 1988] Raymond McKendall and Max Mintz. Robust fusion of location information. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pages 1239–1244, IEEE, Philadelphia, April 1988.
- [Zeytinoglu and Mintz, 1984] Mehmet Zeytinoglu and Max Mintz. Optimal fixed size confidence procedures for a restricted parameter space. *Ann. Statist.*, 12(3):945–957, September 1984.

[Zeytinoglu and Mintz, 1988] Mehmet Zeytinoglu and Max Mintz. Robust fixed size confidence procedures for a restricted parameter space. *Ann. Statist.*, 16(3):1241–1253, September 1988.

Non-Monotonic Decision Rules for Sensor Fusion

Raymond McKendall and Max Mintz*

GRASP Laboratory

Department of Computer and Information Science

University of Pennsylvania

Philadelphia, PA 19104-6389

Abstract

This article describes non-monotonic estimators of a location parameter θ from a noisy measurement $Z = \theta + V$ when the possible values of θ have the form $\{0, \pm 1, \pm 2, \dots, \pm n\}$. If the noise V is Cauchy, then the estimator is a non-monotonic step function. The shape of this rule reflects the non-monotonic shape of the likelihood ratio of a Cauchy random variable. If the noise V is Gaussian with one of two possible scales, then the estimator is also a non-monotonic step function. The shape this rule reflects the non-monotonic shape of the likelihood ratio of the marginal distribution of Z given θ under a least-favorable prior distribution.

1 Introduction

This article describes non-monotonic estimators in decision problems motivated by sensor fusion. It finds minimax rules under zero-one (0) loss for the location parameter θ in two problems of the fusion paradigm $Z = \theta + V$. The statistical background for this research is reviewed in the article *Statistical Decision Theory for Sensor Fusion* [McKendall, 1990b] of these Proceedings, which also defines notation and terminology.

The first problem is a standard-estimation problem in which $\theta \in \{0, \pm 1, \pm 2, \dots, \pm n\}$, for a given integer n , and in which the noise V has the standard Cauchy distribution. A motivation for these assumptions is extension of the results of [Zeytinoglu and Mintz, 1984] and [McKendall, 1990a] that assume the distribution of V has a monotone likelihood ratio.¹ The noise distributions in most practical applications do not have monotone likelihood ratios; the Cauchy distribution is a simple distribution that does not have a monotone likelihood ratio. The minimax rule for this problem is a non-monotonic function. In contrast, the decision rules corresponding

to a noise distribution with a monotone likelihood ratio are monotonic functions.

The second problem is a robust-estimation problem in which $\theta \in \{-1, 0, 1\}$ and the noise V has either the $\mathcal{N}(0, \sigma_1^2)$ or the $\mathcal{N}(0, \sigma_2^2)$ distribution. If the maximum allowable scale is not too large, the robust-estimation problems of [Zeytinoglu and Mintz, 1988] and [McKendall, 1990a] reduce to standard-estimation problems. The underlying distributions in these problems have a monotone likelihood ratio (in the location parameter), and so their minimax rules are monotonic. In contrast, this problem has a non-monotonic minimax rule because the maximum scale is too large. (A similar problem in which the possible locations are an interval has a randomized minimax rule. [Martin, 1987].)

Section 2 discusses the standard-estimation problem with the Cauchy noise distribution. Section 3 discusses the robust-estimation problem with uncertain noise distribution. The results listed here are a synopsis of results in [McKendall, 1990a], which gives the underlying analysis and the proofs.

2 Cauchy Noise Distribution

This section constructs a ziggurat minimax rule δ^* for the location parameter in a standard-estimation problem $(\Theta_n, \Theta_n, L_0, Z)$ in which Z has a Cauchy distribution. A ziggurat decision rule is a non-monotonic step function with range Θ_n . The non-monotonicity of δ^* reflects the non-monotonicity of the likelihood ratio of a Cauchy distribution. The range of δ^* reflects the structure of the zero-one (e) loss function.

Section 2.1 reviews the Cauchy distribution. Section 2.2 summarizes the main results. The remaining sections develop these results in more detail. Their organization follows the strategy for finding a minimax decision rule by finding a Bayes equalizer rule. Section 2.3 defines ziggurat decision rules. Section 2.4 discusses Bayes analysis of a ziggurat decision rule. Sections 2.5, 2.6, and 2.7 give the risk analysis of a ziggurat decision rule. Section 2.8 combines the conclusions of this chapter to find an admissible minimax estimator.

2.1 Cauchy Distribution

A continuous random variable V has the Cauchy distribution with location parameter μ and unit scale, written

*Acknowledgement. Navy Contract N0014-88-K-0630, AFOSR Grants 88-0244, 88-0296; Army/DAAL 03-89-C-0031PRI; NSF Grants CISE/CDA 88-22719, IRI 89-06770; and the Dupont Corporation.

¹A random variable Z with a density function $f_Z(\cdot|\theta)$, for $\theta \in \Theta$, has a monotone likelihood ratio if the ratio $f_Z(\cdot|\theta_1)/f_Z(\cdot|\theta_2)$ is non-decreasing for all $\theta_1 > \theta_2$.

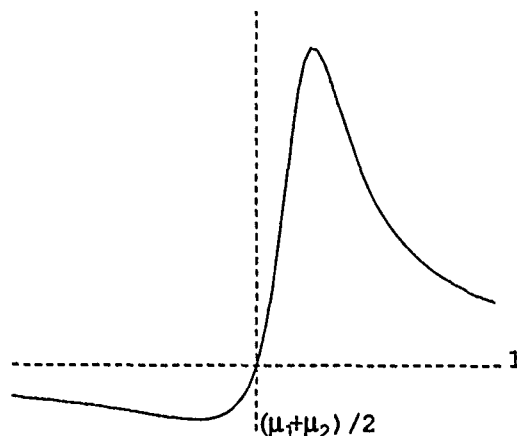


Figure 1: A likelihood ratio $f(\cdot|\mu_1)/f(\cdot|\mu_2)$ of a Cauchy distribution

$V \sim \mathcal{C}(\mu, 1)$, if its density function f is

$$f(v|\mu) = \frac{1}{\pi(1 + (v - \mu)^2)}.$$

The distribution function of a $\mathcal{C}(\mu, 1)$ random variable is

$$F(v|\mu) = \frac{1}{\pi} \arctan\left(\frac{v - \mu}{1}\right) + \frac{1}{2}.$$

The $\mathcal{C}(0, 1)$ distribution is the *standard Cauchy* distribution. An important property of a Cauchy distribution is that it does not have a monotone likelihood ratio. Figure 1 illustrates the shape of these ratios.

2.2 Introduction

This section introduces and summarizes the results through an example. In particular, it shows how to construct a minimax rule δ^* and a least-favorable probability function π^* on Θ_n for the standard-estimation problem $(\Theta_n, \Theta_n, L_0, Z)$ in which $n = 2$ and F is the $\mathcal{C}(0, 1)$ distribution. The general results have arbitrary n .

The decision rule δ^* , defined by figure 2, is the *ziggurat decision rule* over a partition $\{x_i\}_0^5$ of \mathbb{R}^+ onto Θ_2 . It is an even, non-monotonic step function with range Θ_2 and with steps of unit height occurring at points of $\{x_i\}$. The points x_1 and x_2 are chosen so that δ^* is an equalizer rule. The points x_3 and x_4 and the positive probability function π^* are constructed from x_1 and x_2 so that δ^* is Bayes against π^* . Consequently, the rule δ^* is admissible and minimax, and the probability function π^* is least favorable.

The partition $\{x_i\}$ requires solution of the *ziggurat-equalizer equations*:

$$2h_0(y_1) = g_1(y_1) + h_1(y_2) = g_3(y_3)$$

The functions g_i and h_i are these:

$$\begin{aligned} g_i(x) &:= F(x - i) + F(i - \mu_i(x)), & i = 1, 2 \\ h_i(x) &:= F(\mu_{i+1}(x) - i) + F(x - i), & i = 0, 1 \end{aligned}$$

The function μ_i is this:

$$\mu_i(x) := \begin{cases} i - \frac{1}{2} & \text{if } x = i - \frac{1}{2} \\ \frac{(i - \frac{1}{2})x - (i - \frac{1}{2})^2 + v_1^2}{x - (i - \frac{1}{2})} & \text{if } x \neq i - \frac{1}{2} \end{cases}$$

$$v := \frac{1}{2}\sqrt{5}$$

These equations have unique solution y_1, y_2 such that

$$y_1 \in (\frac{1}{2}, \frac{1}{2} + v_1) \text{ and } y_2 \in (\frac{3}{2}, \frac{3}{2} + v_1).$$

Furthermore, $y_1 < y_2$. (The solution may be computed numerically by the Newton-Raphson method.) The partition $\{x_i\}$ is defined in terms of this solution:

$$\begin{aligned} x_0 &:= 0 \\ x_1 &:= y_1 \\ x_2 &:= y_2 \\ x_3 &:= \mu_2(y_2) \\ x_4 &:= \mu_1(y_1) \\ x_5 &:= \infty \end{aligned}$$

This partition is a μ_i -constrained partition of \mathbb{R}^+ .

The probability function π^* is this:

$$\begin{aligned} \pi^*(\pm 1) &= \pi^*(0)/\rho(1) \\ \pi^*(\pm 2) &= \pi^*(0)/(\rho(1)\rho(2)) \end{aligned}$$

The factors $\rho(\pm l)$ connect π^* to $\{x_i\}$ and thus to δ^* :

$$\rho(l) := \frac{f_Z(x_l|l)}{f_Z(x_l|l-1)} =: 1/\rho(-l)$$

The probability function π^* is positive and unique.

2.3 Ziggurat Decision Rule

This section defines and illustrates ziggurat decision rules. A ziggurat rule is specified in terms of a partition of \mathbb{R}^+ .

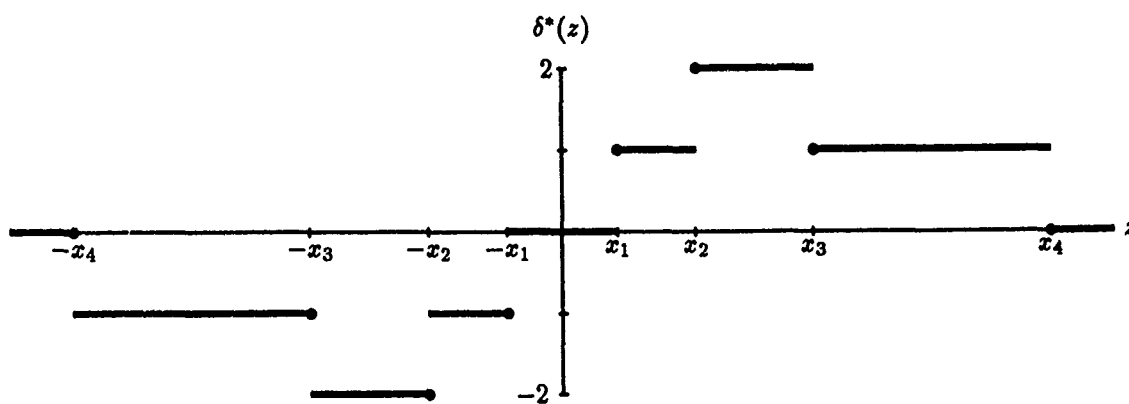


Figure 2: Ziggurat decision rule δ^*

Notation: \mathcal{I}_p^q For integers $p \leq q$, the notation \mathcal{I}_p^q means the integers from p to q . For example, $\mathcal{I}_0^p = \{0, 1, \dots, p\}$.

Definition: partition of \mathfrak{R}^+ A partition² of \mathfrak{R}^+ is a set of points $\{x_i\}_0^{p+1}$ such that $x_0 = 0$, $x_{p+1} = \infty$, and $x_{i+1} > x_i$ for $i \in \mathcal{I}_0^p$. Such a partition is abbreviated as $\{x_i\}$.

Example 2.1 A partition of \mathfrak{R}^+ with $p = 4$ is

$$\{x_i\}_0^5 = \{0, 0.617, 1.912, 4.536, 11.209, \infty\}. \square$$

Remark A particular partition of \mathfrak{R}^+ is specified by the points x_i , $i \in \mathcal{I}_1^p$. The specification of x_0 and x_{p+1} is implicit.

Definition: ziggurat decision rule Let $\{x_i\}_0^{2n+1}$ be a partition of \mathfrak{R}^+ . The ziggurat decision rule δ over $\{x_i\}$ onto Θ_n is this:

$$\delta(z) := \begin{cases} i & \text{if } x_i \leq z < x_{i+1}, \quad i = 0, \dots, n \\ n-i & \text{if } x_{n+i} \leq z < x_{n+i+1}, \quad i = 1, \dots, n \\ -\delta(-z) & \text{if } z \leq 0 \end{cases}$$

Example 2.2 Let $n = 2$. Define δ :

$$\delta(z) := \begin{cases} 0 & \text{if } 0 \leq z < x_1 \\ u & \text{if } x_1 \leq z < x_2 \\ 2u & \text{if } x_2 \leq z < x_3 \\ u & \text{if } x_3 \leq z < x_4 \\ 0 & \text{if } x_4 \leq z \\ -\delta(-z) & \text{if } z < 0 \end{cases}$$

Then δ is the ziggurat decision rule over the partition $\{0, x_1, x_2, x_3, x_4, \infty\}$ onto Θ_2 . \square

Remark The ziggurat rule over $\{x_i\}_0^{2n+1}$ steps between $i-1$ and i at x_i and between i and $i-1$ at x_{2n+1-i} , $i \in \mathcal{I}_1^n$.

Remark The term *ziggurat* loosely describes the shape of the rule over \mathfrak{R}^+ . A ziggurat is a terraced pyramid.

²This definition differs from the set-theoretic definition of some contexts.

2.4 Bayes Rule

Notation

Bayes analysis of a ziggurat rule for a decision problem $(\Theta_n, \Theta_n, L_0, Z)$ in which Z has a Cauchy distribution requires μ_i -constrained partitions of \mathfrak{R}^+ .

Notation $\xi_i := (i - \frac{1}{2}, i - \frac{1}{2} + v)$

Definition: μ_i -constrained partition of \mathfrak{R}^+ A μ_i -constrained partition of \mathfrak{R}^+ is a partition $\{x_i\}_0^{2n+1}$ of \mathfrak{R}^+ such that for all $i \in \mathcal{I}_1^n$,

$$x_i \in \xi_i$$

and

$$x_{2n+1-i} = \mu_i(x_i).$$

Example 2.3 A μ_i -constrained partition of \mathfrak{R}^+ has the following structure:

$$\{0, x_1, x_2, \dots, x_{n-1}, x_n, \mu_n(x_n), \mu_{n-1}(x_{n-1}), \dots, \mu_2(x_2), \mu_1(x_1), \infty\}$$

Furthermore, $x_i \in \xi_i$. \square

Example 2.4 Let $n = 2$. Define x_1, x_2, x_3, x_4 :

$$x_1 := 0.617, x_2 := 1.912, x_3 := 4.536, x_4 := 11.209.$$

Note that $x_1 \in \xi_1$ and $x_2 \in \xi_2$:

$$\frac{1}{2} < x_1 < \frac{1}{2} + \frac{1}{2}\sqrt{5} = 1.618$$

$$\frac{3}{2} < x_2 < \frac{3}{2} + \frac{1}{2}\sqrt{5} = 2.618$$

Verify that $x_3 = \mu_2(x_2)$ and $x_4 = \mu_1(x_1)$. Therefore, $\{0, x_1, x_2, x_3, x_4, \infty\}$ is a μ_i -constrained partition of \mathfrak{R}^+ . \square

Remark Let $\{x_i\}_0^{2n+1}$ be a μ_i -constrained partition of \mathfrak{R}^+ . The ziggurat rule over $\{x_i\}$ steps between $i-1$ and i at x_i and between i and $i-1$ at $\mu_i(x_i)$, $i \in \mathcal{I}_1^n$.

Remark Let $f_Z(\cdot|i) \sim \mathcal{C}(i, 1)$, where i is an integer. The function μ_i satisfies the identity

$$\frac{f_Z(\mu_i(x)|i+e)}{f_Z(\mu_i(x)|i-e-1)} = \frac{f_Z(x|i+e)}{f_Z(x|i-e-1)}, \quad \forall x \in \mathfrak{R}.$$

This is the functional definition of μ_i . Bayes analysis motivates this definition. The algebraic definition of μ_i is derived from the functional definition.

Main Result

Proposition 1 shows that to any ziggurat decision rule δ over a μ_i -constrained partition of \mathbb{R}^+ , there corresponds a positive probability function π on Θ_n such that δ is Bayes against π .

Proposition 1 Assume $F \sim C(0, 1)$. Let $\{x_i\}_0^{2n+1}$ be a μ_i -constrained partition of \mathbb{R}^+ . Let π be the even, positive probability function on Θ_n such that for all $l \in I_1^n$,

$$\pi(l-1) = \rho(l)\pi(l).$$

The ziggurat decision rule over $\{x_i\}$ onto Θ_n is Bayes against π .

Example 2.5 Let $n = 2$. Let $\{x_i\}_0^5$ be the μ_i -constrained partition of \mathbb{R}^+ given in example 2.4:

$$\{x_i\} = \{0, .617, 1.912, 4.536, 11.209, \infty\}$$

Let δ be the ziggurat decision rule over $\{x_i\}$ onto Θ_2 :

$$\delta(z) = \begin{cases} 0 & \text{if } 0 \leq z < 0.617 \\ 1 & \text{if } 0.617 \leq z < 1.912 \\ 2 & \text{if } 1.912 \leq z < 4.536 \\ 1 & \text{if } 4.536 \leq z < 11.209 \\ 0 & \text{if } 11.209 \leq z \\ -\delta(-z) & \text{if } z < 0 \end{cases}$$

Then δ is Bayes against some positive probability function on Θ_2 . □

Example 2.6 Consider example 2.5. The conditions of proposition 1 for a probability function π on Θ_2 are these:

$$\begin{aligned} \pi(0) &= \rho(1)\pi(1) \\ \rho(1) &:= \frac{f_Z(x_1|1)}{f_Z(x_1|0)} = \frac{f(0.617-1)}{f(0.617)} = 1.204 \\ \pi(1) &= \rho(2)\pi(2) \\ \rho(2) &:= \frac{f_Z(x_2|2)}{f_Z(x_2|1)} = \frac{f(1.912-2)}{f(1.912-1)} = 1.818 \end{aligned}$$

Also, $\pi(-1) = \pi(1)$ and $\pi(-2) = \pi(2)$. Hence:

$$\begin{aligned} \sum_{\theta} \pi(\theta) &= \pi(0) \left(1 + \frac{2}{\rho(1)} + \frac{2}{\rho(1)\rho(2)} \right) \\ &= 3.575\pi(0) \end{aligned}$$

Thus π assigns these probabilities:

$$\begin{aligned} \pi(0) &= 0.280 \\ \pi(\pm 1) &= 0.232 \\ \pi(\pm 2) &= 0.128 \end{aligned}$$

Therefore, the ziggurat decision rule over $\{x_i\}_0^5$ onto Θ_2 is Bayes against the probability function π on Θ_2 . □

Example 2.7 The probability function π of proposition 1 is given by the following equations: For all $l \in I_1^n$,

$$\pi(\pm l) = \left(\prod_{k=1}^l \frac{f_Z(x_k|k)}{f_Z(x_k|k-1)} \right)^{-1} \pi(0),$$

where

$$\pi(0) = \left[1 + 2 \sum_{l=1}^n \left(\prod_{k=1}^l \frac{f_Z(x_k|k)}{f_Z(x_k|k-1)} \right)^{-1} \right]^{-1} \quad \square$$

Remark In proposition 1, the restriction to a μ_i -constrained partition of \mathbb{R}^+ and the conditions on the probability function are necessary for the decision rule to minimize the posterior expected loss.

2.5 Risk Function

Proposition 2 gives the risk function of a ziggurat decision rule over a μ_i -constrained partition of \mathbb{R}^+ .

Proposition 2 Let $\{x_i\}_0^{2n+1}$ be a μ_i -constrained partition of \mathbb{R}^+ , and let δ be the ziggurat decision rule over $\{x_i\}$ onto Θ_n .

$$\begin{aligned} R(0, \delta) &= 2h_0(x_1) \\ R(\pm i, \delta) &= g_i(x_i) + h_i(x_{i+1}), \quad i \in I_1^{n-1} \\ R(\pm n, \delta) &= g_n(x_n) \end{aligned}$$

Example 2.8 Let $n = 3$. Let $\{x_i\}_0^7$ be a μ_i -constrained partition of \mathbb{R}^+ , and let δ be the ziggurat decision rule over $\{x_i\}$ onto Θ_3 .

$$\begin{aligned} R(0, \delta) &= 2h_0(x_1) \\ R(\pm u, \delta) &= g_1(x_1) + h_1(x_2) \\ R(\pm 2u, \delta) &= g_2(x_2) + h_2(x_3) \\ R(\pm 3u, \delta) &= g_3(x_3) \quad \square \end{aligned}$$

2.6 Ziggurat-Equalizer Equations

Equating the expressions $R(\theta, \delta)$ over $\theta \in \Theta_n$ to find a ziggurat equalizer rule leads to the *ziggurat-equalizer equations*. These are n equations in n unknowns y_1, \dots, y_n . For $n = 1$, the ziggurat-equalizer equation is

$$2h_0(y_1) = g_1(y_1).$$

For $n \geq 2$, the ziggurat-equalizer equations are

$$2h_0(y_1) = g_1(y_1) + h_1(y_{l+1}) = g_n(y_n), \quad l \in I_1^n.$$

Example 2.9 The ziggurat-equalizer equations for $n = 2$ are these:

$$2h_0(y_1) = g_1(y_1) + h_1(y_2) = g_2(y_2).$$

The ziggurat-equalizer equations for $n = 3$ are these:

$$2h_0(y_1) = g_1(y_1) + h_1(y_2) = g_2(y_2) + h_2(y_3) = g_3(y_3). \quad \square$$

Proposition 3 states that the ziggurat-equalizer equations have a unique solution y_1, \dots, y_n that has certain properties. Proposition 4 uses this solution to construct an equalizer rule.

Proposition 3 Assume $F \sim C(0, 1)$. The ziggurat-equalizer equations have unique, increasing solution y_1, \dots, y_n with $y_l \in \xi_l$. Furthermore $y_l - y_{l-1} > 1$ for $l \in I_2^n$.

Example 2.10 Let $F \sim C(0, 1)$. The ziggurat-equalizer equations for $n = 3$ and $u = 1$ have the following solution:

$$\begin{aligned} y_1 &= 0.570743 \\ y_2 &= 1.731856 \\ y_3 &= 2.979961 \end{aligned}$$

Here, $y_1 \in (0.5, 0.5 + v_1)$, $y_2 \in (1.5, 1.5 + v_1)$, and $y_3 \in (2.5, 2.5 + v_1)$. Also $y_2 - y_1 > 1$ and $y_3 - y_2 > 1$. □

2.7 Equalizer Rule

Proposition 4 gives a ziggurat equalizer rule.

Proposition 4 Assume $F \sim C(0,1)$. Let y_1, \dots, y_n with $y_i \in \xi_i$ satisfy the ziggurat-equalizer equations. For $i \in \mathcal{I}_1^n$, define

$$x_i := y_i \text{ and } x_{2n+1-i} := \mu_i(y_i).$$

Also, define $x_0 := 0$ and $x_{2n+1} := \infty$. If $\{x_i\}_0^{2n+1}$ is a partition of \mathbb{R}^+ , then the ziggurat decision rule δ over $\{x_i\}$ onto Θ_n is an equalizer rule. Furthermore, if $\{x_i\}$ is a partition of \mathbb{R}^+ , then the common risk of δ is $R_\delta = g_n(x_n)$ and $F(-\frac{1}{2}) < R_\delta < 2F(-\frac{1}{2})$.

Example 2.11 Let $n = 3$. The solution y_1, y_2, y_3 to the ziggurat-equalizer equations specified by the proposition is

$$y_1 = 0.571, y_2 = 1.732, y_3 = 2.980.$$

Let $x_1 := y_1, x_2 := y_2$, and $x_3 := y_3$. Also, define x_4, x_5 , and x_6 as follows:

$$\begin{aligned} x_4 &:= \mu_3(x_3) = 5.104 \\ x_5 &:= \mu_2(x_2) = 6.891 \\ x_6 &:= \mu_1(x_1) = 18.170 \end{aligned}$$

Note that $\{x_i\}$ is a partition of \mathbb{R}^+ :

$$\{x_i\} = \{0, 0.571, 1.732, 2.980, 5.104, 6.891, 18.170, \infty\}.$$

Thus, the ziggurat decision rule over $\{x_i\}$ onto Θ_3 is an equalizer. Its risk is $R_\delta = g_3(x_3)$:

$$\begin{aligned} g_3(x_3) &= F(x_3 - 3) + F(3 - \mu_3(x_3)) \\ &= F(x_3 - 3) + F(3 - x_4) \\ &= 0.635 \end{aligned}$$

Here, $0.352 = F(-\frac{1}{2}) < R_\delta < 2F(-\frac{1}{2})$. \square

Example 2.12 Refer to example 2.5: Verify that $y_1 := 0.617$ and $y_2 := 1.912$ satisfy the ziggurat-equalizer equations for $n = 2$. Thus, since $\{x_i\}$ is a μ_i -constrained constrained partition of \mathbb{R}^+ , the ziggurat rule over $\{x_i\}$ is an equalizer rule. \square

Remark Proposition 3 asserts that x_1, \dots, x_n exist and that $x_i > x_{i-1}, i \in \mathcal{I}_2^n$. There is no guarantee, however, that $\{x_i\}_0^{2n+1}$ is a partition of \mathbb{R}^+ ; it is necessary to verify that $\mu_{i-1}(x_{i-1}) > \mu_i(x_i), i \in \mathcal{I}_2^n$. If $\{x_i\}$ is a partition of \mathbb{R}^+ , then it is a μ_i -constrained partition by construction. Numerical computations suggest that $\{x_i\}$ is in fact a partition of \mathbb{R}^+ , but there is no proof of this conjecture.

2.8 Minimax Rule

Theorem 1 combines the conclusions of this chapter to find an admissible minimax estimator of the location parameter θ for a decision problem $(\Theta_n, \Theta_n, L_0, Z)$ in which Z has a Cauchy distribution.

Theorem 1 Assume $F \sim C(0,1)$. Let y_1, \dots, y_n with $y_i \in \xi_i$ satisfy the ziggurat-equalizer equations. For $i \in \mathcal{I}_1^n$, define

$$x_i := y_i \text{ and } x_{2n+1-i} := \mu_i(y_i).$$

Also, define $x_0 := 0$ and $x_{2n+1} := \infty$. Suppose that $\{x_i\}_0^{2n+1}$ is a partition of \mathbb{R}^+ , and let δ^* be the ziggurat decision rule over $\{x_i\}$ onto Θ_n .

Let π^* be the positive probability function on Θ_n defined by the following conditions: For $i \in \mathcal{I}_1^n$,

$$\pi^*(\pm i) = \left(\prod_{k=1}^i \rho(k) \right)^{-1} \pi^*(0),$$

where

$$\pi^*(0) = \left[1 + 2 \sum_{i=1}^n \left(\prod_{k=1}^i \rho(k) \right)^{-1} \right]^{-1}$$

Then δ^* and π^* have the following properties:

1. δ^* is Bayes against π^* .
2. δ^* is an equalizer rule.
3. δ^* is minimax.
4. δ^* is admissible.
5. π^* is least favorable.

Example 2.13 Refer to example 2.11: The ziggurat decision rule over $\{x_i\}$ onto Θ_3 is an admissible minimax rule. \square

Example 2.14 Refer to examples 2.5 and 2.6: Verify that $y_1 := 0.617$ and $y_2 := 1.912$ satisfy the ziggurat-equalizer equations for $n = 2$, and note that $\{x_i\}$ is a μ_i -constrained constrained partition of \mathbb{R}^+ . Thus δ is minimax and π is least favorable. \square

Corollary 2 In theorem 1, define

$$\tau := F(-\frac{1}{2})/F(\frac{1}{2}).$$

Then

$$F(-\frac{1}{2}) < R_{\delta^*} \leq 1 - \left(1 + 2\tau \frac{1 - \tau^N}{1 - \tau} \right)^{-1}$$

Remark The upper bound of this corollary is better than the upper bound $2F(-\frac{1}{2})$ of proposition 4:

$$1 - \left(1 + 2\tau \frac{1 - \tau^N}{1 - \tau} \right)^{-1} \uparrow 2F(-\frac{1}{2}u) \text{ as } N \uparrow \infty$$

3 Uncertain Noise Distribution

This section constructs a minimax rule for the location parameter in a robust-estimation problem $(\Theta_1 \times \{\sigma_1, \sigma_2\}, \Theta_1, L_0, Z)$ in which the uncertainty class is $\{\mathcal{N}(0, \sigma_1^2), \mathcal{N}(0, \sigma_2^2)\}$. The larger scale σ_2 is large enough that the problem does not reduce to standard-estimation. Examples 3.1 and 3.2 give minimax rules for specific values of the scales. Example 3.3 considers a similar problem in which the scale set has more than two points. The minimax rules of these examples are not monotonic even though the nominal distribution has a monotone likelihood ratio in its location parameter. Examples 3.4 - 3.7 discuss the analysis underlying these results.

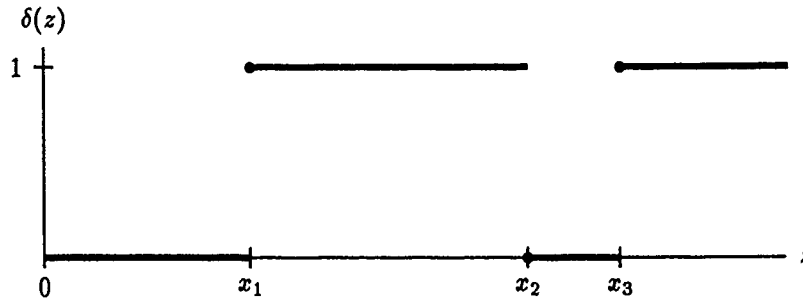


Figure 3: A minimax rule for $(\Theta_1 \times \{\sigma_1, \sigma_2\}, \Theta_1, L_0, Z)$ ($z \geq 0$)

Example 3.1 Let $\sigma_1 := 1$ and $\sigma_2 := 2.5$. Define the decision rule δ^* as follows: this:

$$\begin{aligned} x_1 &:= 1.09833 \\ x_2 &:= 2.59355 \\ x_3 &:= 3.095 \end{aligned}$$

$$\begin{aligned} \pi^*(0, \sigma_1) &:= 0 \\ \pi^*(0, \sigma_2) &:= 0.43414873 \\ \pi^*(\pm 1, \sigma_1) &:= 0.09183446 \\ \pi^*(\pm 1, \sigma_2) &:= 0.19109118 \end{aligned}$$

$$\delta^*(z) := \begin{cases} 0 & \text{if } 0 \leq z < x_1 \\ 1 & \text{if } x_1 \leq z < x_2 \\ 0 & \text{if } x_2 \leq z < x_3 \\ 1 & \text{if } x_3 \leq z \\ -\delta^*(-z) & \text{if } z < 0. \end{cases} \quad (1)$$

(See figure 3.) This rule is a minimax rule for $(\Theta_1 \times \{\sigma_1, \sigma_2\}, \Theta_1, L_0, Z)$.

Let π^* be the following probability function on $\Theta_1 \times \{\sigma_1, \sigma_2\}$:

$$\begin{aligned} \pi^*(0, \sigma_1) &:= 0 \\ \pi^*(0, \sigma_2) &:= 0.40587187 \\ \pi^*(\pm 1, \sigma_1) &:= 0.048166 \\ \pi^*(\pm 1, \sigma_2) &:= 0.24890241 \end{aligned}$$

Then δ^* is a Bayes rule against π^* , and π^* is a least-favorable probability function.

The rule δ^* is almost an equalizer rule over $\Theta_1 \times \{\sigma_1, \sigma_2\}$:

$$\begin{aligned} R((0, \sigma_1), \delta^*) &= 0.26453 \\ R((0, \sigma_2), \delta^*) &= R((\pm 1, \sigma_1), \delta^*) = R((\pm 1, \sigma_2), \delta^*) \\ &= 0.576597 \end{aligned}$$

The risk for the parameter $(0, \sigma_1)$ is less than the equalized risk for the other pairs, and the probability mass for $(0, \sigma_1)$ is zero. \square

Example 3.2 Let $\sigma_1 := 1$ and $\sigma_2 := 2$. The corresponding points x_1, x_2, x_3 are these:

$$\begin{aligned} x_1 &:= 1.09504 \\ x_2 &:= 2.93635 \\ x_3 &:= 3.20822 \end{aligned}$$

Define δ^* by definition (1). Then δ^* is minimax. The corresponding least-favorable probability function π^* is

The risk function is this:

$$\begin{aligned} R((0, \sigma_1), \delta^*) &= 0.271514 \\ R((0, \sigma_2), \delta^*) &= R((\pm 1, \sigma_1), \delta^*) = R((\pm 1, \sigma_2), \delta^*) \\ &= 0.550656 \end{aligned}$$

In this example, too, the risk for the parameter $(0, \sigma_1)$ is less than the equalized risk for the other parameters, and the probability mass for $(0, \sigma_1)$ is zero. \square

Example 3.3 This example extends example 3.2 by allowing the scale set to have more than two points.

Define $\sigma_0 = 0.9073846$. Let Σ be a scale set that includes σ_1, σ_2 , and any finite number of points between σ_0 and σ_1 . Then δ^* is robust minimax for the decision problem $(\Theta_1 \times \Sigma, \Theta_1, L_0, Z)$. The probability function of example 3.2 is extended as follows: If $\sigma \neq \sigma_1$ or $\sigma \neq \sigma_2$, then $\pi^*(\theta, \sigma) := 0$ for all θ . Here, too, δ^* is Bayes against π^* , and π^* is least favorable. \square

Example 3.4 In the standard-estimation problems of [McKendall, 1990a], the likelihood ratio of the sampling density $f_Z(\cdot|\theta)$ is important to Bayes analysis. If Z has a monotone likelihood ratio, for example, the corresponding Bayes rule is monotonic. Alternatively, if Z has a Cauchy distribution, the non-monotonic shape of a Bayes rule mimics the non-monotonic shape of a Cauchy likelihood ratio. In this robust-estimation problem, however, it is the likelihood ratio of the *marginal* density of Z given θ under the least-favorable distribution π^* , denoted $\beta_Z(\cdot|\theta)$, that is important to Bayes analysis:

$$\beta_Z(z|\theta) := \sum_{\sigma} f_Z(z|(\theta, \sigma)) \pi(\theta, \sigma), \quad z \in \mathbb{R}$$

Figure 4 plots a likelihood ratio of $\beta_Z(\cdot|\theta)$ for the robust-estimation problem of example 3.1. The non-monotonic shape of δ^* mimics the shape of this ratio. \square

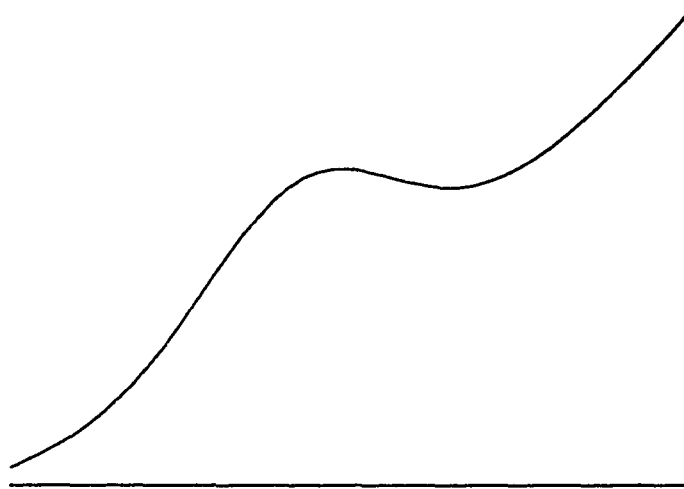


Figure 4: A likelihood ratio of $\beta_Z(\cdot|\theta)$

Example 3.5 The probability function π^* of example 3.1 or 3.2 satisfies the following linear system of equations:

$$\beta_Z(x_i|1) = \beta_Z(x_i|0), \quad i = 1, 2, 3$$

$$\sum_{\theta} \sum_{\sigma} \pi^*(\theta, \sigma) = 1$$

Define y_0, y_1, y_2 , and y_3 :

$$y_0 := \pi^*(0, \sigma_1)$$

$$y_1 := \pi^*(0, \sigma_2)$$

$$y_2 := \pi^*(1, \sigma_1)$$

$$y_3 := \pi^*(2, \sigma_2)$$

The equations are these ($i = 1, 2, 3$):

$$\begin{aligned} \frac{1}{\sigma_1} f\left(\frac{x_i}{\sigma_1}\right) y_0 + \frac{1}{\sigma_2} f\left(\frac{x_i}{\sigma_2}\right) y_1 \\ - \frac{1}{\sigma_1} f\left(\frac{x_i - 1}{\sigma_1}\right) y_2 - \frac{1}{\sigma_2} f\left(\frac{x_i - 1}{\sigma_2}\right) y_3 = 0 \end{aligned}$$

$$y_0 + y_1 + 2y_2 + 2y_3 = 1$$

When x_1, x_2 , and x_3 are known, these are four equations in four variables.

These constraints on the probability function are analogous to those of proposition 1. \square

Example 3.6 The results of examples 3.1, and 3.2 are computed from the following nonlinear system of equations with the assumption that $\pi^*(0, \sigma_1) = 0$ (or $y_0 = 0$):

$$y_1 + 2y_2 + 2y_3 = 1$$

$$\beta_Z(x_i|1) = \beta_Z(x_i|0), \quad i = 1, 2, 3$$

$$R((1, \sigma_j), \delta^*) = R(0, \sigma_2), \delta^*), \quad j = 1, 2$$

These are six equations in the six unknowns $x_1, x_2, x_3, y_1, y_2, y_3$. It must be verified for any solution that $x_1 \leq x_2 \leq x_3$, that y_1, y_2 , and y_3 are non-negative, that δ^* is Bayes against π^* , and that $R((0, \sigma_1), \delta^*) \leq R((0, \sigma_2), \delta^*)$. \square

Example 3.7 This example lists the risk function of a decision rule δ^* of definition (1).

$$R((0, \sigma), \delta^*) = -2F(x_1/\sigma) + 2F(x_2/\sigma) + 2F(-x_3/\sigma)$$

$$R((1, \sigma), \delta^*) = F((x_1 - 1)/\sigma) - F((x_2 - 1)/\sigma) + F((x_3 - 1)/\sigma)$$

$$R((-1, \sigma), \delta^*) = R((1, \sigma), \delta^*) \square$$

References

- [Martin, 1987] K.E. Martin. *Randomized Robust Confidence Procedures*. PhD thesis, Department of Systems Engineering, University of Pennsylvania, December 1987.
- [Martin and Mintz, 1984] K.E. Martin and M. Mintz. Randomized robust confidence procedures. In *Proceedings of the Twenty-Second Annual Allerton Conference on Communication, Control, and Computing*, pages 309-317, University of Illinois, October 1984.
- [McKendall, 1990a] R. McKendall. *Minimax Estimation of a Discrete Location Parameter for a Continuous Distribution*. PhD dissertation. GRASP Lab technical report MIS-CS-90-28, Department of Computer and Information Science, University of Pennsylvania, May 1990.
- [McKendall, 1990b] R. McKendall. Statistical decision theory for sensor fusion. In these Proceedings, October 1990.
- [Zeytinoglu and Mintz, 1984] M. Zeytinoglu and M. Mintz. Optimal fixed sized confidence procedures for a restricted parameter space. *The Annals of Statistics*, 12(3):945-957, September 1984.
- [Zeytinoglu and Mintz, 1988] M. Zeytinoglu and M. Mintz. Robust optimal fixed sized confidence procedures for a restricted parameter space. *The Annals of Statistics*, 16(3):1241-1253, September 1988.

Real-Time Vergence Control for Binocular Robots

Thomas J. Olson*

olson@cs.virginia.edu

Department of Computer Science
University of Virginia
Charlottesville, Virginia 22903

David J. Coombs

coombs@cs.rochester.edu

Department of Computer Science
University of Rochester
Rochester, New York 14627

Abstract

In binocular systems, *vergence* is the process of adjusting the angle between the eyes (or cameras) so that both eyes are directed at the same world point. Its utility is most obvious for foveate systems such as the human visual system, but it is a useful strategy for non-foveate binocular robots as well. This paper discusses the vergence problem and outlines a general approach to vergence control, consisting of a control loop driven by an algorithm that estimates the vergence error. As a case study, this approach is used to verge the eyes of the Rochester Robot in real time. Vergence error is estimated with the cepstral disparity filter. The demonstration system uses a PD controller in cascade with the error estimator. Empirical measurements of the performance of both the disparity estimator and the overall system are presented.

1 Introduction

Recently a significant amount of work in computer vision has focused on the problems of acting, behaving systems, and in particular on how "active vision" differs from analysis of static scenes or vision with fixed cameras [Bajcsy, 1986, Aloimonos *et al.*, June 1987, Ballard, 1989]. In many cases, giving a vision system the ability to move around in its environment simplifies many previously intractable problems. Since the summer of 1988 the Rochester vision group has been working to develop an integrated facility for the study of vision, AI and systems issues related to active vision. Briefly, the facility consists of an industrial robot arm bearing a custom-built "head". The head has two CCD television

cameras which can be moved together in altitude (pitch) and independently in azimuth (yaw). The head, arm and cameras are connected to a pipelined image processor, a workstation and a set of large-scale parallel processors.

A major goal of our research is the development of a real-time gaze control system. We believe that the robot must be able to maintain fixation on world points or change fixation from one world point to another with only minimal direction from high level "cognitive" faculties. To this end, we are developing quasi-reflexive gaze control mechanisms that maintain fixation on smoothly moving targets while compensating for egomotion, and make saccadic movements to targets selected by higher level processes. We envision the gaze control mechanisms forming a layered, modular control structure along the lines described by Brooks [Brooks, 1986], although we suspect that more sensor fusion may be required than has been employed in systems of this type in the past. The details of the control structure and module interactions are a current research topic [Brown, 1989, Coombs, 1989], but preliminary work has identified some promising approaches to the various subproblems [Brown *et al.*, 1988].

This paper describes the design, implementation and performance of a module responsible for controlling the vergence angle of the cameras. The next section discusses vergence in the abstract, presenting reasons for verging and issues that any vergence control system must address. This discussion leads to a general strategy for vergence control, described in Section 3. Sections 4, 5 and 6 describe the application of this vergence control strategy to the problem of controlling vergence on the Rochester Robot, and present empirical results on the performance of the error estimator and the overall vergence system.

2 The Vergence Problem

The *vergence angle* of a binocular system is the angle between the optic axes of its eyes or cameras. The vergence angle, baseline (or inter-ocular distance) and gaze direction of a binocular system determine a particular fixation point, as shown in figure 1. Narrowly speaking, the function of the vergence system is to control the distance from the cameras to the fixation point along some specified gaze direction. In most cases the motivation for vergence is to keep the fixation point near some tar-

*This material is based on work supported by the U.S. Army Engineering Topographic Laboratories under research contract no. DACA76-85-C-0001, by NSF research grants nos. DCR-8602958 and IRI-8903582, by NIH Public Health Service research grant no. 1 R01 NS22407-01, and by ONR research contract no. N00014-82-K-0193. In addition, this work was supported by the NSF under Institutional Infrastructure grant CDA-8822724, and by a Dean's Research Initiation grant from the School of Engineering and Applied Science of the University of Virginia. The government has certain rights in this material.

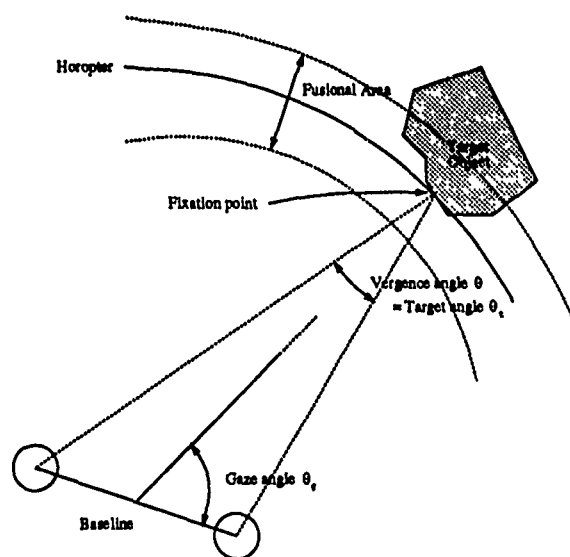


Figure 1: The goal of vergence is to keep the eyes or cameras fixed on a common world point or visual target, independent of changes in gaze angle and target distance. The result of fixating a target is that the object lies near the *horopter*, which is the set of world points whose disparity is zero.

get object. Thus, the vergence problem can be defined as that of controlling the vergence angle to keep the fixation depth appropriate for the current gaze target. Since the target vergence angle is directly related to target depth, any sensory cue to depth or depth changes may be useful to the vergence system. The most commonly used cues are disparity and focus error, but other depth cues (such as motion, texture, shading, *etc.*) can also be used, as can information about depth changes (measured or predicted self motions, dilations or contractions of the visual field, and so on).

Vergence is one aspect of the larger problem of gaze control, which involves control of the gaze angle and focal depth as well. The larger problem can be broken down functionally into the subproblems of *gaze stabilization* and *gaze shift*. Stabilization involves maintaining fixation on a possibly moving visual target from a possibly moving gaze platform. Gaze shifts, usually called *saccades*, transfer fixation from one visual target to another. Vergence control must meet different demands in each of these activities. During stabilization, a change in the target position relative to the observer produces a smooth change in the desired vergence angle. A saccade transfers the fixation point almost instantaneously from one visual target to another, producing a step change in the desired vergence angle.

2.1 Related Work

The recent surge of interest in active vision has produced a growing body of literature on vergence and gaze control for robotic vision systems. Clark and Ferrier [1988] built a gaze control system based on the model described in [Robinson, 1987]. The system acquires and tracks

white and black blobs using the first few moments and intensity value of each object. The mechanical design of their head decouples the control of the gaze and vergence angles. The gaze angle is controlled by rotating the head about its neck, and the cameras are verged symmetrically by a mechanical linkage. Vergence has recently been used cooperatively with focus and stereopsis for surface reconstruction [Abbott and Ahuja, 1988] and active exploration of the environment [Krotkov, 1989]. It has demonstrated advantages in both robustness of results and increased speed in stereoscopic processing.

2.2 Why Verge?

Vergence control is clearly necessary for systems whose sensors have non-uniform resolution, such as the human visual system. Vergence movements allow humans to register objects of interest in the high-resolution central regions of both eyes, so that the greatest possible amount of information can be extracted. An argument can be made that non-uniform resolution is ultimately necessary in order to provide both high resolution and a wide field of view [Tsotsos, 1987], but most current robot vision systems do not have foveas. However, vergence has many advantages even for systems without foveas.

Mathematical Simplification: Fixating an object of interest puts points on the object near the optic axis in both eyes. In some cases this permits the use of simplifying assumptions (*e.g.* replacing perspective projection with orthography) that make analysis significantly easier (*e.g.* [Ballard and Ozcanlarli, 1988]).

Facilitating Stereo Fusion: By definition, the fixation point has a stereoscopic disparity of zero, and points nearby tend to have small disparities. This makes it possible to use stereo algorithms that accept only a limited range of disparities. Such systems can be very fast, and are amenable to hardware implementation [Mead and Mahowald, 1988].

Useful Coordinate Systems: As Ballard [1989] argues, having a unique fixation point at the intersection of the visual axes defines a coordinate system that is related as much to the object being observed as it is to the observer. It is thus a step in the direction of an object-centered coordinate system.

Disparity-based Segmentation: On the assumption that gaze will normally be directed toward objects of interest, it may be appropriate for binocular agents to ignore features at large disparities. That is, disparity may be used to filter objects that are not currently of interest out of the scene. Figure 2 shows two examples of this sort of filtering.

3 A General Strategy for Vergence Control

At the most abstract level, any solution to the vergence problem will have three major components: a *sensory system* that determines how the current vergence angle differs from the ideal, a *controller* that generates a response to the errors, and a *motor system* that executes

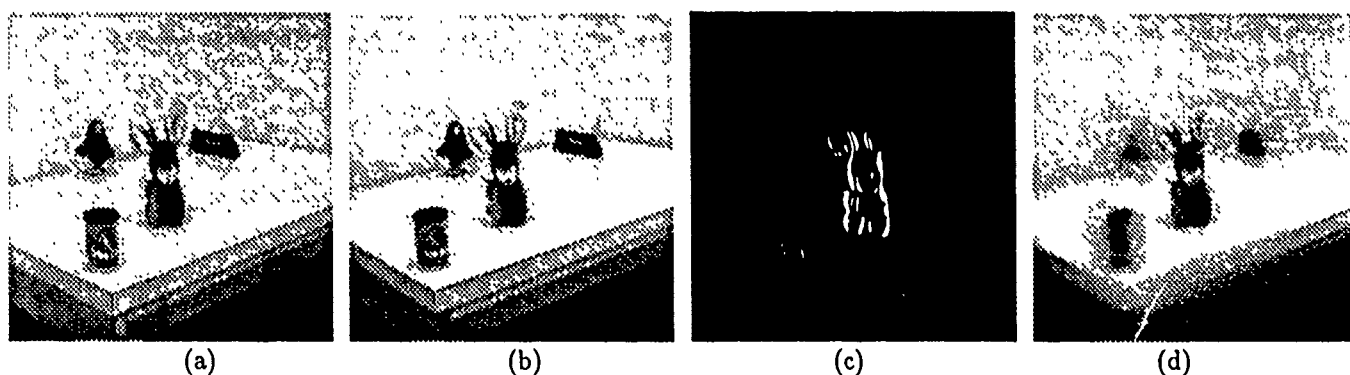


Figure 2: Using disparity to filter a scene. Images (a) and (b) above show left and right camera views of a scene containing objects at several disparities. Two simple filters can be used to pass more spatial detail in regions of the scene that have smaller disparities. Image (c) is the result of "ANDING" the vertical edge images to suppress edges that have non-zero disparity, leaving an edge image that is dominated by objects at the horopter. Image (d) is produced by "ANDING" bandpass images (laplacian pyramids [Burt and Adelson, April 1983]) instead of edge images, which has the effect of suppressing high frequency information associated with objects at high disparities.

the controller's commands. This section discusses general considerations in the design and use of these components.

3.1 Motor System

The quality of the motor system or plant is determined by how quickly and faithfully it translates control signals into changes in vergence angle. Current generation CCD cameras and motor controllers make it relatively easy to move the cameras quickly. Care is required, however, to insure that the camera mounting is able to tolerate the stresses generated by rapid eye movements. The large accelerations required for saccadic movements can cause "ringing", i.e. vibrations that persist after the motors have come to a stop. Avoiding these problems involves mechanical engineering considerations that are beyond the scope of this paper, so we will not discuss them further.

3.2 Controller

A critical parameter in the design of a vergence control system is the nature of the input signal - how the target angle θ_t changes with time. Our expectations are based on the known characteristics of human eye movements [Yarbus, 1967], and on the general view of gaze control described in Section 2. That is, we expect eye movements to consist of intervals of smooth pursuit or fixation punctuated by discontinuous jumps (saccades).

The two types of expected changes in target angle differ in fundamental ways. During pursuit and fixation, changes in target angle are determined by the dynamics of observer and object motion. The input to the control loop during pursuit and fixation will be smooth, with finite second derivatives and small first derivatives. During a saccade the input signal will behave quite differently. A saccade can produce a step change in the desired vergence angle, as well as a discontinuity in its temporal derivative.

The fact that there are two distinct types of changes in desired vergence angle suggests a need for two modes

of control. The normal operating mode should be optimized for the smooth, continuous changes expected during pursuit and fixation. Saccadic movements should replace the smooth movements of the normal control loop with open-loop moves at maximum speed to the new desired vergence angle.

3.3 Error Estimator

In order to keep the eyes verged on a target, the vergence system must measure the current vergence error (and, perhaps, its derivatives.) The most important source of this information is the visual system, but other sources may also be useful. We have already noted the possibility of predicting the error that will result from a saccade to a target of known depth. Vergence changes due to self motion can also be taken into account, either by making predictions based on planned, voluntary head movements, or by sensing head accelerations via the vestibular system (as in the human vestibulo-ocular reflex.) However, vision is the only source of information for target motion, and visual cues also provide the ultimate measure of vergence performance. The rest of this section, therefore, is restricted to consideration of visual error estimators.

A number of different types of visual information are available for estimating vergence error. One feature that is correlated with desired vergence angle under ordinary conditions is blur, which has been used cooperatively with vergence and stereo to construct depth maps [Abbott and Ahuja, 1988, Krotkov, 1989]. Any depth cue can be used if the absolute vergence angle of the system is known, because desired vergence angle is a function of target distance.

The most useful visual cue to vergence error, however, is binocular disparity. The mapping from disparity to vergence error is particularly simple, and (unlike monocular depth cues) does not require knowledge of the absolute vergence angle of the system. Reliable disparity estimates can be computed more easily and quickly than depth estimates, permitting shorter processing de-

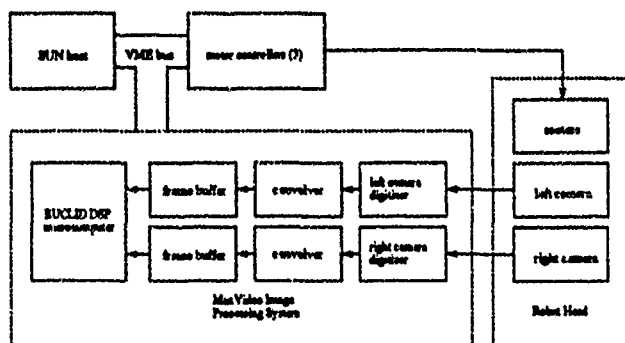


Figure 3: Hardware configuration used for vergence control experiments on the Rochester Robot.

lays and simpler control strategies.

The types of disparity estimators that have been used for estimating stereoscopic depth are poorly suited to controlling vergence. For real-time vergence what is needed is a simple algorithm that estimates a single disparity in a fixed amount of time. This narrows the field to image processing methods such as various forms of cross-correlation.

4 Vergence on the Rochester Robot

The general considerations discussed in the preceding sections formed the basis for the vergence system used on the Rochester Robot. This section and the two that follow describe the motor, sensory and control components of the system, and discuss its performance as measured in the laboratory.

We begin by summarizing those aspects of the Rochester Robot's cameras, motor system, and computing resources that affected the design of the vergence system. A more detailed description of the robot and laboratory resources is given in [Brown *et al.*, 1988]. Figure 3 shows a block diagram of those parts of the system that are involved in vergence control.

As illustrated in Figure 4, each of the robot's two cameras is panned from side to side by its own motor. A third motor serves to pitch the cameras up and down, *i.e.*, to rotate them about their baseline. These three motors can generate saccades with peak speeds of more than 400 degrees per second. Reduction gearing gives the cameras theoretical angular resolutions of 1/278 degree in yaw and 1/2500 degree in pitch. Gear lash reduces the resolution to an unknown degree, but camera positioning is still accurate to substantially better than the angle subtended by one pixel with the 16mm lenses that are normally used.

The host computer for the robot commands the motors via intelligent stepping motor controllers that allow the control program to issue commands in terms of absolute position, relative position, velocity or velocity profile. The ability to issue buffered velocity commands enables the control program to generate smooth movements without paying constant attention to the motors.

The EUCLED digital signal processing microcomputer included in the MaxVideoTM image processing system

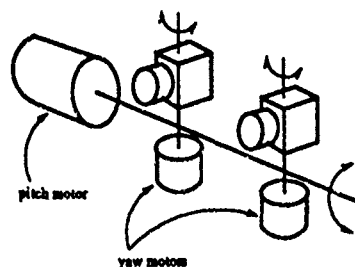


Figure 4: Mechanical design of the rochester robot head.

was used for estimating disparity. The EUCLED computer is based on the ADSP-2100 digital signal processor, which is optimized for operations such as convolution, finite impulse response filtering and Fast Fourier Transforms.

5 Error Estimation

The vergence error estimator is based on disparity, since (as argued in Section 3) disparity is the most direct and reliable measure of vergence error. One approach to disparity estimation would have been to use the MaxVideoTM convolution/correlation hardware to compare central patches of one image to the other image. However, previous attempts in our lab to use that approach for tracking had encountered many difficulties. Instead the disparity estimator, previously described in [Olson and Potter, 1989], is based on the cepstral filter [Yeshurun and Schwartz, 1989]. The cepstrum can be viewed as correlation with an adaptive pre-filter, and is thus related to phase correlation [Kuglin and Hines, 1975]. This section describes the basic operation and performance of the cepstral disparity estimator.

5.1 Measuring Disparity with the Cepstral Filter

The *cepstrum* of a signal is the Fourier transform of the log of its power spectrum. It was developed by Bogert *et al.* [Bogert *et al.*, 1963] as a tool for analyzing signals containing echoes. Such signals can be modeled as an original signal $S(t)$ convolved with a train of impulses, *i.e.*,

$$R(t) = S(t) * (\delta(t) + a_0\delta(t - t_0) + a_1\delta(t - t_1) + \dots)$$

where $*$ denotes convolution. Taking the log of the power spectrum transforms the received signal into a sum of two terms, one of which depends only on $S(t)$ and the other of which is a combination of distorted sinusoids with frequencies related to t_0, t_1 , etc. If the cepstrum of $S(t)$ does not overlap the frequencies of the echo terms, conventional linear filtering techniques can be used to extract the values of the echo delays.

Recently Yeshurun and Schwartz [Yeshurun and Schwartz, 1989] developed a way of using the two-dimensional cepstrum as a disparity estimator. Sample windows of size $h \times w$ are extracted from each image and spliced together along one edge to produce an image of size $h \times 2w$. Assuming that the right and left

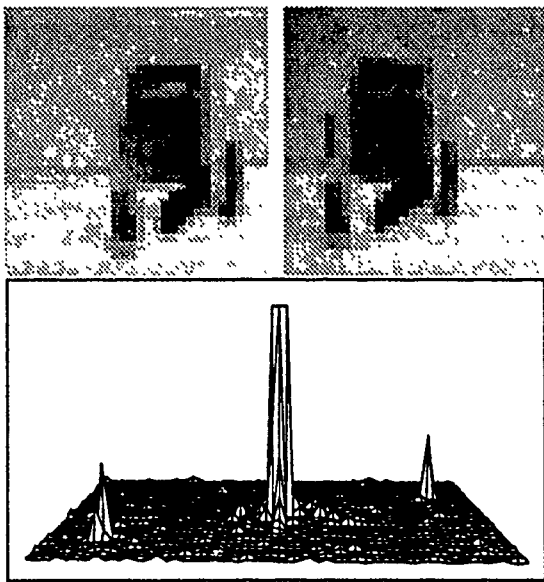


Figure 5: Cepstral disparity estimator sample input and outputs. At top are two 32×32 subsampled images taken by the left and right cameras of the Rochester Robot. Below is a surface plot of the power spectrum of the cepstral filter implementation. The small peaks at left front and right rear give the disparity.

images differ only by a shift, the spliced image may be thought of as an original image at $(0,0)$ plus an echo at $(w+d_h, d_v)$, where d_h and d_v are the horizontal and vertical disparities. The periodic term in the cepstrum of such a signal will have fundamental frequencies of $w+d_h$ horizontally and d_v vertically. These are high frequencies relative to the window size. The image-dependent term, by contrast, will be composed of much lower frequencies, barring pathological images. Thus the cepstrum of the signal will usually have clear, isolated peaks at $(\pm(w+d_h), \pm d_v)$.

5.2 Implementation

In order to obtain an acceptable temporal sampling rate, the cepstral error estimate is computed using the MaxVideo image processing system. The images are formed by the robot's CCD cameras, which are synchronized so that right and left images reflect the state of the world at the same point in time and become available simultaneously. The video signals are digitized and convolved with anti-aliasing filters (Gaussian, $\sigma = 2.5$ pixels) before being stored in frame buffer memory. The EUCLID DSP microprocessor then extracts 32×32 sample windows from the central 256×256 regions of each image, and computes the cepstral disparity estimate. The final implementation computes the cepstral disparity estimate for 32×32 windows in approximately 51 milliseconds, not including digitization time or the 8 ms required to acquire the VME bus and read the sample arrays from the frame buffer. Figure 5 shows a sample input and a plot of the cepstral output.

Although the implementation described above is ade-

quate for some purposes, its accuracy is limited to about ± 15 minutes of arc by the coarse quantization of the sample windows. The current implementation obtains sub-pixel resolution by first finding the peak pixel value in the cepstral output region and then interpolating to better localize the disparity peak. The interpolated value is a weighted centroid of the peak pixel and its left and right neighbors, the weights being chosen adaptively to discount the contribution of the neighbors when their values are comparable to background noise.

5.3 Performance of the cepstral disparity estimator

The implementation of the cepstral estimator described above was tested in the laboratory on several scenes. For each test, the robot was directed to face the scene and the cameras were manually adjusted to the correct vergence angle for the scene. Taking that angle as the home or zero-disparity position, the test program then swept the cameras over a range of vergence angles. At each position it recorded the actual disparity (represented by the difference between the commanded position and the home position), and the disparity reported by the cepstral estimator running on the EUCLID DSP computer.

Figure 6 shows results from a typical laboratory scene. The estimator fails badly at one end of its range because at that point the target object is no longer visible in both sample windows. Within a ± 7 degree range, however, performance is good. The RMS error over that range is 1.31 image pixels (4.44 arc minutes). Most of this error is due to a problem with the empirically determined constant multiplier used to convert disparity in pixels to disparity in degrees. Because the axes of rotation of the cameras do not pass through their nodal points, the nodal points undergo some translation when the cameras rotate. This means that the conversion constant has a small dependence on the depth of the target. The target used in this example is not at the optimal distance for the constant being used, leading to a small systematic error in disparity estimates. This systematic error could be removed by taking target depth (inferred from current eye position and approximate disparity) into account when converting from pixels to degrees. This has not been necessary to date, because small errors at large disparities have a negligible effect on the performance of the control loop. High accuracy is important only at disparities near zero, where errors or discontinuities can cause the target angle to overshoot or oscillate around the desired value.

The limiting performance of the cepstral estimator can be determined by comparing the measured disparity values to a best-fit straight line. For the data set shown here, this yields an RMS error of 0.64 pixels (2.24 arc minutes). In other words, the estimate is potentially accurate to two thirds the width of an image pixel. This is quite good, particularly in view of the fact that the cepstral implementation subsamples by a factor of eight. Relative to its sample window resolution, the cepstral RMS error is on the order of one twelfth of a pixel.

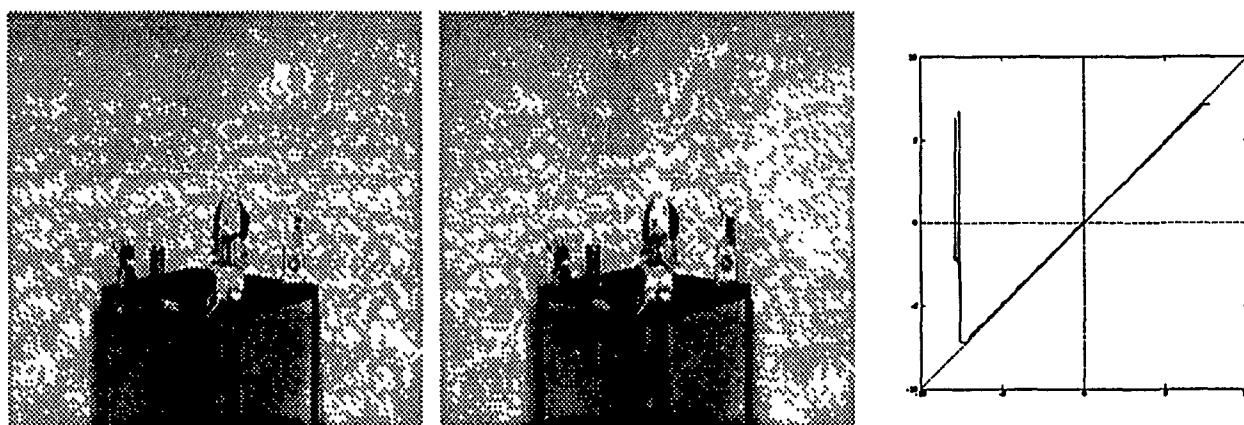


Figure 6: Performance of the cepstral disparity estimator on a typical laboratory scene. At left and center are left and right camera images, taken near zero disparity. At right is a plot of measured versus actual disparity.

6 Control

The goal of the vergence system is to generate smooth eye movements that correct the vergence error. The vergence control loop consists of three stages: digitization, error estimation, and error correction. Digitization is done under control of the SunTM host and takes between one and two RS-170 frame times. Once the images are available in the frame store, the Sun signals EUCLID to extract the images from the frame buffers and estimate the disparity. This process takes approximately 59 milliseconds, after which EUCLID places the disparity estimate in a known location in shared memory and issues an interrupt to signal completion. The Sun converts the pixel disparity to angular coordinates by multiplying it by an empirically determined constant, and executes the control law to issue the appropriate velocity command to the eye motors. The loop consistently takes 3 frame times to complete. Thus, the system achieves a servo rate of 10 Hz.

6.1 The Controller

The vergence system uses a proportional-derivative (PD) controller (*e.g.*, see [Dorf, 1980]) in cascade with the eye motor in a feedback loop. (Although the target and actual vergence angle are continuous variables, since the entire system under our control is digital or presents digital interfaces we model the system discretely.) The summation node that produces the error signal represents the process of estimating vergence angle error from the disparity of binocular images acquired from the cameras. The controller gains were chosen empirically to obtain slightly underdamped response, resulting in a small overshoot in the step response. The system controls the velocities of the motors to achieve smooth responses to smoothly varying stimuli, controlling the accelerations explicitly would require more computational expense and constant attention of the Sun host.

6.2 Performance

The demonstration system's responses to step and sinusoidal stimuli were measured in the lab. Representative

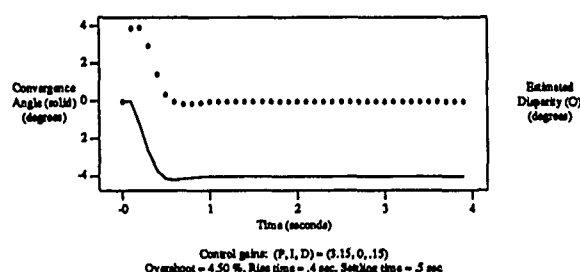


Figure 7: Response to a step in disparity: The convergence angle is drawn solid, and the disparity estimate is plotted in circles. *Rise time* is the earliest time the response reaches 90% of its final (steady-state) value, and *settling time* is the earliest time the response stays within 5% of its final value. Note that the sample interval is 0.1 seconds.

camera angle traces of step and sinusoidal responses are shown in Figures 7 and 8. Figure 9 summarizes the system's response to sinusoidal stimuli of frequencies up to 2 Hz. For ease of measurement, the system was not run in the normal mode of compensating for half the error with each camera, but rather one camera alone was moved to correct the entire error and the angle of this camera was recorded.

The step stimulus was produced by manually misconverging the "verging" camera prior to starting the system. The same effect could be achieved by misconverging the camera in the dark and then switching on the lights suddenly at time 0. In the response (Figure 7), observe the single time step (0.1 second) latency in detecting the disparity. As a consequence of this delay, the estimated disparity is seen to lag behind the camera's convergence angle, even though this disparity estimate provides the error signal that drives the vergence system. The small overshoot results from slight underdamping.

Analogous to the step stimulus, the sinusoidal stimuli were generated by rotating the non-verging camera

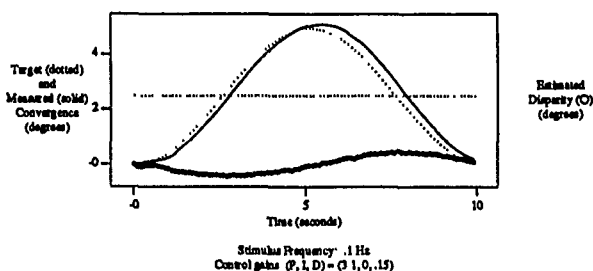


Figure 8: Response to sinusoidal disparity stimulus: The target convergence angle is shown dotted and the measured response is drawn solid. The midline of the target signal is shown dotted for reference. The disparity estimate is plotted in circles.

sinusoidally. If the verging camera is held still, this generates a sinusoidally oscillating disparity signal. Thus, the target vergence angle was defined by the angle of the non-verging (stimulating) camera. The system's response to a 0.1 Hz stimulus is plotted in Figure 8.

The vergence responses to sinusoidal stimuli were measured for frequencies ranging from 0.05 to 2 Hz. The gain and the phase shift of the system's responses are summarized in the Bode plot of Figure 9. The system's behavior suggests that it may be a second order system. However, the constant time delay seems to produce a linear phase shift, since a constant time delay contributes proportionately more to phase shift at higher frequencies.

7 Conclusion

We have argued that vergence is important for active vision systems, and have discussed general issues in the design of vergence control systems. We have also described in detail the application of these ideas to develop a real-time vergence control system for the Rochester Robot.

The error estimator for the vergence system is a variant of the cepstral disparity estimator of Yeshurun and Schwartz [1989]. The estimator has been shown to be capable of remarkable accuracy, in the best case achieving an RMS error of a small fraction of a pixel. It is simple enough that with a small investment in special hardware it can be computed at speeds comparable to the video frame rate. The cepstral method of disparity estimation can be shown to be equivalent to autocorrelation of images that have been adaptively enhanced to sharpen their autocorrelation functions. It is thus closely related to phase correlation.

The demonstration system uses a position controller that generates smooth vergence camera movements in response to smooth changes in the desired vergence angle. This system also responds reasonably (but suboptimally) to step changes in target vergence angle. Optimal response to step stimuli could be achieved by saccadic vergence movements.

Vergence responses have yet to be integrated with other gaze controls. Future work in the area of gaze control will concentrate on the questions. What gaze

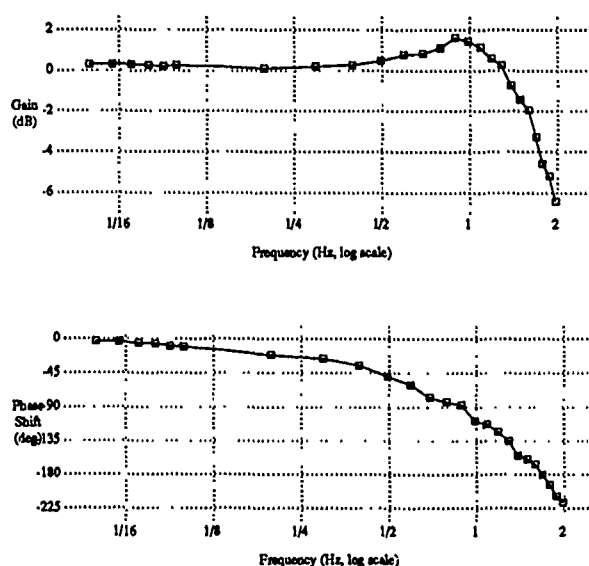


Figure 9: Bode plot: The gain (dB) and phase shift (degrees) of the vergence responses are shown for sinusoidal stimuli of frequencies ranging from 0.05 to 2 Hz. $\text{Gain (dB)} = 20 * \log_{10}(\frac{\text{response amplitude}}{\text{stimulus amplitude}})$ and the phase shift is the difference in the phase angle of the two signals.

controls are primitive, what cues serve them best, and how do visually-mediated controls interact? Interaction of controls can be simple (say by preemption), or more complex (with controls aware of and cooperating with the actions of other controls [Brown, 1989, Coombs, 1989]). The former approach requires breaking down the controls into either orthogonal, non-interacting primitives or being content to have one control acting at a time. The latter approach requires more sophisticated modeling of the effects of interaction.

Another area for future exploration concerns the use of camera systems that offer vergence and fixation as reliable primitives. One obvious application is the support of stereo systems with limited fusional ranges [Olson, 1990]. More generally, systems that fixate must choose appropriate targets for the task they are performing. Thus gaze control at the highest level can be viewed as a resource management problem, in which limited sensory and computing hardware must be allocated so as to maximize the usefulness of the recovered information [Rimey and Brown, 1990].

Acknowledgments

Robert Potter wrote an early version of the vergence loop. Dana Ballard, Chris Brown and Randal Nelson provided leadership and advice, both technical and editorial. Mike King and Larry Snyder shared their expertise in biological eye movement systems and control systems. The comments of Ray Rimey, Mike Swain, and Lambert Wixson improved the presentation. Dave Tilley

and Ray Rimey enhanced the MaxVideoTM programming environment with Zebra and Zed [Tilley, 1990] and Tim Becker helped make the PUMATM software usable.

References

- [Abbott and Ahuja, 1988] A. Lynn Abbott and Narendra Ahuja. Surface reconstruction by dynamic integration of focus, camera vergence, and stereo. In *International Conference on Computer Vision*. IEEE, 1988.
- [Aloimonos et al., June 1987] J. Aloimonos, I. Weiss, and A. Bandopadhyay. Active vision. In *Proc. 1st Int'l. Conf. on Computer Vision*, pages 35-54, London, June 1987.
- [Bajcsy, 1986] Ruzena Bajcsy. Passive perception vs. active perception. In *Proc. IEEE Workshop on Computer Vision*, Ann Arbor, 1986.
- [Ballard and Ozcandarli, 1988] Dana H. Ballard and Altan Ozcandarli. Eye movements and visual cognition: Kinetic depth. In *International Conference on Computer Vision*. IEEE, December 1988.
- [Ballard, 1989] Dana H. Ballard. Reference frames for animate vision. In *International Joint Conference on Artificial Intelligence*. AAAI, 1989.
- [Bogert et al., 1963] B. P. Bogert, M. J. R. Healy, and J. W. Tukey. The quefrency analysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking. In M. Rosenblatt, editor, *Proc. Symp. Time Series Analysis*, pages 209-243. John Wiley and Sons, 1963.
- [Brooks, 1986] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, pages 14-23, April 1986.
- [Brown et al., 1988] Christopher Brown, Dana Ballard, Timothy Becker, Roger Gans, Nathaniel Martin, Thomas Olson, Robert Potter, Raymond Rimey, David Tilley, and Steven Whitehead. The rochester robot. Technical Report 257, University of Rochester, Computer Science Department, 1988. Christopher M. Brown (Editor).
- [Brown, 1989] Christopher M. Brown. Prediction in gaze and saccade control. Technical Report 295, University of Rochester, Computer Science Department, 1989.
- [Burt and Adelson, April 1983] Peter J. Burt and Edward H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532-540, April 1983.
- [Clark and Ferrier, 1988] James Clark and Nicola Ferrier. Modal control of an attentive vision system. In *International Conference on Computer Vision*. IEEE, 1988.
- [Coombs, 1989] David J. Coombs. Gaze stabilization for animate vision. Thesis Proposal, December 1989.
- [Dorf, 1980] Richard C. Dorf. *Modern control systems*. Addison-Wesley, 3rd edition, 1980.
- [Krotkov, 1989] Eric Paul Krotkov. *Active computer vision by cooperative focus and stereo*. Springer-Verlag, 1989.
- [Kuglin and Hines, 1975] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. In *Proc. IEEE Int'l Conf. on Cybernetics and Society*, pages 163-165, 1975.
- [Mead and Mahowald, 1988] Carver A. Mead and M. A. Mahowald. A silicon model of early visual processing. *Neural Networks*, 1:pp. 91-97, 1988.
- [Olson and Potter, 1989] Thomas J. Olson and Robert D. Potter. Real-time vergence control. In *Proc. Computer Society Conference on Computer Vision and Pattern Recognition*, pages 404-409, San Diego, June 1989.
- [Olson, 1990] Thomas J. Olson. Stereopsis for fixating systems. Technical Report in preparation, University of Virginia Department of Computer Science, 1990.
- [Rimey and Brown, 1990] R. D. Rimey and C. M. Brown. Selective attention as sequential behavior: Modelling eye movements with an augmented hidden markov model. Technical Report 327, University of Rochester, Computer Science Department, February 1990.
- [Robinson, 1987] David Robinson. Why visuomotor systems don't like negative feedback and how they avoid it. In Michael Arbib and Allen Hanson, editors, *Vision, Brain and Cooperative Computation*. MIT Press, 1987.
- [Tilley, 1990] David G. Tilley. Zebra for maxvideo: an application of object oriented microprogramming to register level devices. Technical Report 315, University of Rochester, Computer Science Department, 1990.
- [Tsotsos, 1987] John K Tsotsos. A 'complexity level' analysis of vision. In *Proc. First International Conference on Computer Vision*, pages 825-834, London, June 1987.
- [Yarbus, 1967] A. L. Yarbus. *Eye Movements and Vision*. Plenum Press, New York, 1967.
- [Yeshurun and Schwartz, 1989] Yehezkel Yeshurun and Eric Schwartz. Cepstral filtering on a columnar image architecture: A fast algorithm for binocular stereo segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7), July 1989.

Sequential Decision Making for Active Perception

Thomas Dean* Theodore Camus Jak Kirman

Department of Computer Science
Brown University, Box 1910, Providence, RI 02912

Abstract

There are many perceptual tasks in robotics that can be facilitated by planning sequences of perceptual actions. We present an approach to sequential decision making well suited to handling the sort of planning required to direct movement for active perceptual tasks such as object recognition or target tracking. The approach is based on Bayesian decision theory, and relies on encoding the underlying sequential decision problem in terms of a compact network model (the so-called *Bayes network* or *influence diagram* model) for which the evaluation problem is well understood. We illustrate our approach using a specific problem, involving a mobile robot tracking a moving target and continuously reporting on the target's position with respect to a global map.

Introduction

Recently, researchers have become interested in perceptual tasks that allow an agent to actively select a set of sensors and sensor views [Bajcsy, 1988, Ballard, 1989]. Certain problems that are ill posed or computationally difficult given a single data set are well posed and computationally straightforward given an opportunity to select multiple data sets [Aloimonos *et al.*, 1987]. In this paper, we are concerned with the decision making required for guiding this selection process.

A simple image understanding problem might require an agent to select from among several possible views, taking into account the cost of obtaining those views and the benefits to be gained from the resulting information in terms of the agent's ability to discriminate among a set of models in a library of such models [Hager, 1988]. In more complicated problems, it may be useful for an agent to consider sequences of actions. Planning is the process of generating and evaluating

alternative sequences of actions in terms of their predicted consequences so as to choose among them.

In this paper, we provide a probabilistic approach to sequential decision making for active perceptual tasks. Our approach relies on an existing model for reasoning about time and action [Dean and Kanazawa, 1989, Kanazawa and Dean, 1989]. The model for time and action is a special case of Pearl's [1988] *Bayes networks* and Howard and Matheson's [1984] *influence diagrams*. We evaluate our model using a variant of an algorithm due to Lauritzen and Spiegelhalter [1988].

Planning and its associated temporal reasoning requirements provide ample opportunities for combinatorial explosion [Chapman, 1987, Dean and Boddy, 1988]. To be of practical use, a planning solution must control the potential for combinatorial explosion through the use of carefully designed representations. By constructing our decision model in terms of the agent's perceptual capabilities and not in terms of some existing abstraction that bears little resemblance to how the agent perceives its world, we are able to keep the combinatorics to a manageable level.

As a methodological aside, problems of the sort that arise in active perception provide an excellent opportunity for planning researchers to test current methods and explore new ones. In general, robotics applications impose interesting constraints involving time pressure and uncertainty, and they force researchers to be more careful in terms of representational commitments. Active perception requires that we address the problem of extracting and interpreting sensor data to support whatever representational commitments we have made.

In the next section, we consider a particular application for active perception that will be used to illustrate our approach to sequential decision making.

Mobile Target Localization

The application that we have chosen to demonstrate our approach involves a mobile robot navigating and tracking moving targets in a cluttered environment. The robot is provided with sonar and rudimentary vision. The moving target could be a person or another mobile

*This work was supported in part by a National Science Foundation Presidential Young Investigator Award IRI-8957601 with matching funds from IBM, and by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Air Force Office of Scientific Research under Contract No. F49620-88-C-0132.

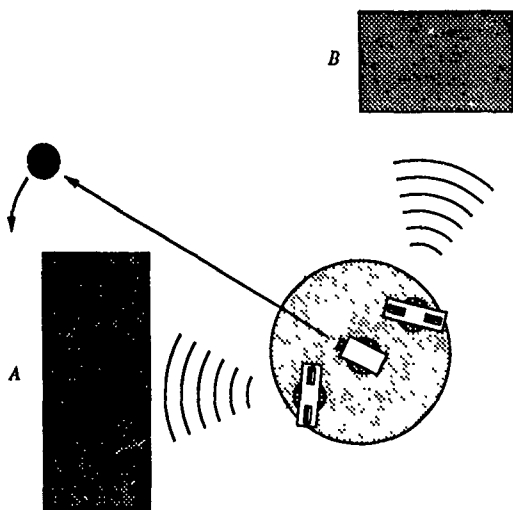


Figure 1: Tracking in a cluttered environment

robot. The mobile base consists of a holonomic (turn-in-place) synchro-drive robot equipped with a CCD camera mounted on a pan-and-tilt head, and two sonar arrays, each of which is mounted on a separate rotating platform. The two sonar arrays and the camera are independently controlled, with a separate process responsible for each of the motor controllers (one for each rotating platform and one for the two motors driving the pan-and-tilt head).

The robot's task is to detect and track moving objects, reporting their location in the coordinate system of a global map. Initially, the robot systematically explores its surroundings scanning for moving objects. Once it detects such an object, it is required to track the object and report on its location. The environment consists of one floor of an office building. The robot is supplied with a floor plan of the office showing the position of permanent walls and major pieces of furniture such as desks and tables. Smaller pieces of furniture, potted plants and other assorted clutter constitute obstacles that the robot has to detect and avoid.

We assume that there is error in the robot's movement requiring it to continually estimate its position with respect to the floor plan so as not to become lost. Position estimation (*localization*) is performed by having the robot track *beacons* corresponding to walls and corners and then use these beacons to reduce error in its position estimate. By keeping track of several beacons and ensuring that the set of beacons at one point in time overlaps significantly with those at nearby points in time, the robot can essentially "triangulate" itself through the world with very little accumulated error [Levitt *et al.*, 1987, Leonard and Durrant-Whyte, 1989].

Localization and tracking are frequently at odds with

one another. A particular localization strategy may reduce position errors while making tracking impossible, or improve tracking while losing registration with the global map. The robot shown in Figure 1 should move more toward the obstacle marked B than toward the obstacle marked A in order to anticipate the target being occluded by A, despite the fact that moving toward A would probably improve localization. The trick is to balance the demands of localization against the demands of tracking.

Planning and Perception

For the mobile target localization (MTL) problem described above, we assume two basic sensory capabilities: sonar for rudimentary feature extraction and depth estimation, and vision for optical flow calculation to detect and track moving objects.

The independently controlled sonar arrays—each array consisting of a pair of Polaroid ultrasonic transducers—are aligned with flat surfaces to obtain accurate estimates of the robot's distance from such surfaces. Alignment is achieved by using feedback to minimize the difference between the readings from two sonars in a given array. This simple method allows us to deal with the sort of specular surfaces commonly found in office environments. Flat walls, convex and concave corners, and other geometric features can be detected and distinguished reliably by continuously tracking the sonar signals returned from these features [Leonard and Durrant-Whyte, 1989]. The methods are accurate enough to reliably detect doorframes around closed doors. Speed in sequencing the transducers and interpreting the results is still a major problem in making these techniques practical.

The primary sensory capability used for tracking will be optical flow, the point-wise description of motion in a 3-D world as projected in a 2-D image. We assume that we are concerned with rigid objects, and that in general the motion of one point in an image is the same as that of adjacent points in the image, excepting discontinuities such as edges. A recent parallel optical-flow algorithm implements this assumption by tracking the motion of a template window of pixels centered at the pixel in question, rather than the single pixel alone, for every pixel in the image [Bülthoff *et al.*, 1989]. The motion of this template window is determined by the best match between its original position in the first image and all possible positions of the pixel in the second image, typically limited to a maximum displacement of plus or minus some small number of pixels. The best match is taken to be the displacement whose unweighted sum of the absolute differences of the intensity values between the corresponding pixels in the respective templates is a minimum. This best match represents the optical flow of the given pixel, and is repeated for each point to yield the optical-flow field. Despite robust performance on natural images, this algorithm

is unsuitable for practical real-time operation due to its extensive hardware requirements, such as a massively parallel supercomputer. We employ a fast serial implementation of the algorithm using pyramid techniques which is practical [Camus, 1990].

Once we have our low-level output in the form of an optical-flow field, we must somehow extract the motion of our moving target. Although there will be noise in the flow field (manifested as spurious velocities for stationary points), we assume that our moving object is represented in the field as a relatively large cohesive entity of coherent motion, whereas any noise present is randomly distributed. To transform the optical-flow field (which contains thousands of velocity vectors, one per pixel) into a tractable representation, we simply subsample the optical-flow field, resulting in a coarser representation of the optical flow (simple averaging can be used). Vectors corresponding to the moving object are coalesced into larger, more manageable areas, while random noise cancels itself out. The final result is a very coarse (4×4) optical-flow field, with each large block representing a large area of homogeneous motion, presumably corresponding to our target. This gives us both the two-dimensional position of the object as well as its velocity in the image plane, which is sufficient to center the target in the camera's field of view.

Once the tracked object is centered in the field of view, a sonar array is pointed in the direction of the target to recover the distance to the target. From the distance and angular information, we can compute the position of the target relative to the robot. Human targets present diffuse surfaces for sonar imaging, allowing for reasonably accurate distance estimates.

A Model for Action and Perception

In this section, we provide a decision model for the MTL problem described earlier. To specify this model, it will be convenient to quantize the space in which the robot and its target are embedded. A natural quantization can be derived from the robot's sensory capabilities.

The robot's sonar sensors enable it to recognize specific classes of beacons (e.g., walls, concave and convex corners) as well as the distance and orientation of the beacons relative to the robot. It is possible to tessellate the world into regions such that the same beacons are detectable anywhere within a given region. This tessellation of the world provides a set of locations \mathcal{L} corresponding to the regions that are used to encode the location of both the robot and its target. Our decision model includes two variables S_T and S_R , where S_T represents the location of the target and ranges over \mathcal{L} , and S_R represents the location and orientation of the robot and ranges over an extension of \mathcal{L} including orientation information specific to each type of location. For example, in a hallway, the orientation might be specified as one of only two values, while in a T junction or in open space a wider range of orientations will likely

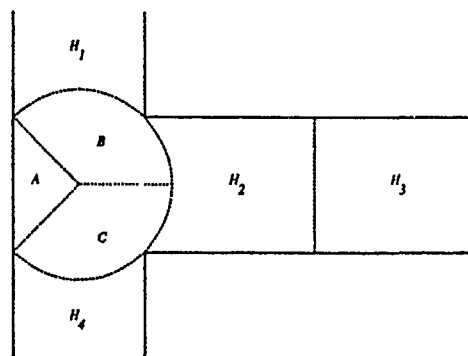


Figure 2: Tessellation of a T-junction

be appropriate. This spatial representation scheme has two important properties. First, the mapping between the information returned by the sensors and the spatial representation is straightforward, thereby simplifying the process of quantifying the probabilistic model. Second, the sizes of S_R and S_T are kept to a minimum by taking into account the robot's sensory limitations, thereby reducing the amount of computation required to evaluate the decision model.

For any particular instance of the MTL problem, we assume that a geometric description of the environment is provided in the form of a CAD model. Given this geometric description and a model for the robot's sensors, we generate \mathcal{L} , S_R , and S_T . Figure 2 shows how a T junction might be tessellated given the beacon tracking system. Notice that the hallways leading into the junction are composed of pieces of space, H_1 , H_2 , H_3 , and H_4 . In all but H_2 , the only beacons visible are the two opposite walls of the hallway. In H_2 , the leftmost wall provides a third beacon. The junction itself is divided into three regions. Region A is distinguished by three beacons: the leftmost wall and the two convex corners. The other two regions of the junction have a wall and one corner; in each case the furthest corner beacon may not be stable due to irregular sonar reflections.

A *Bayes network* is a directed graph $G = (V, E)$. The vertices in V correspond to random variables and are often referred to as *chance nodes*. The edges in E define the causal and informational dependencies between the random variables. In the model described in this paper, chance nodes are discrete valued variables that encode states of knowledge about the world. Let Ω_C be the set of discrete values of a chance node C . There is a probability distribution $\Pr(C = \omega, \omega \in \Omega_C)$ for each node. If the chance node has no predecessors then this is its marginal probability distribution; otherwise, it is a conditional probability distribution dependent on the states of the immediate predecessors of C in G .

The model described here involves a specialization of Bayes networks called *temporal belief networks* [Dean and Kanazawa, 1989]. Given a set of discrete variables,

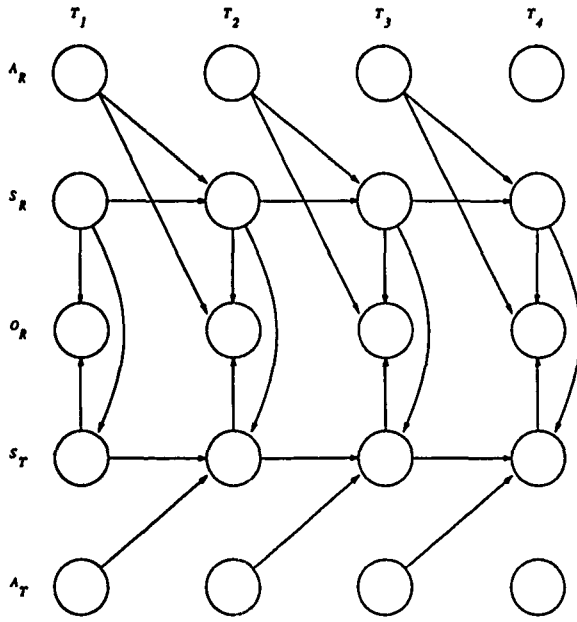


Figure 3: Probabilistic model for the MTL problem

\mathcal{X} , and a finite ordered set of time points, T , we construct a set of chance nodes, $\mathcal{C} = \mathcal{X} \times T$, where each element of \mathcal{C} corresponds to the value of some particular $x \in \mathcal{X}$ at some $t \in T$. Let \mathcal{C}_t correspond to the subset of \mathcal{C} restricted to t . The temporal belief networks discussed in this paper are distinguished by the following Markov property:

$$\Pr(\mathcal{C}_t | \mathcal{C}_{t-1}, \mathcal{C}_{t-2}, \dots) = \Pr(\mathcal{C}_t | \mathcal{C}_{t-1}).$$

Let S_R and S_T be variables ranging over the possible locations of the robot and the target respectively. Similarly, let A_R and A_T be variables ranging over the actions available to the robot and target. At any given point in time, the robot can make certain observations regarding its position with respect to the global map and the target's position with respect to the robot; O_R is a variable ranging over sets of such observations. In a more complete description, we would have separate observation variables for each distinct sensory modality.

Figure 3 shows a temporal belief network for $\mathcal{X} = \{S_R, S_T, A_R, A_T, O_R\}$ and $T = \{T_1, T_2, T_3, T_4\}$. To quantify the model shown in Figure 3, we have to provide distributions for each of the variables in $\mathcal{X} \times T$. We assume that the model does not depend on time, and, hence, we need only provide one probability distribution for each $x \in \mathcal{X}$. For instance, the conditional probability distribution for S_T ,

$$\Pr(\langle S_T, t \rangle | \langle A_T, t-1 \rangle, \langle S_T, t-1 \rangle, \langle S_R, t \rangle),$$

is the same for any $t \in T$. The numbers for the probability distributions can be obtained by experimentation without regard to any particular global map. A useful marginal distribution describing the actions of the

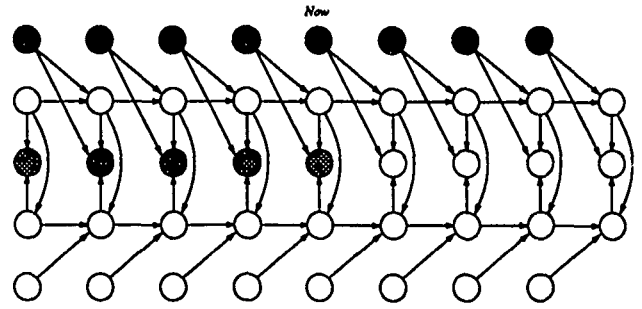


Figure 4: Evidence and action sequences

target, A_T , may be difficult to obtain, but we expect that a fairly simple model for the behavior of mobile targets in spatially restricted environments will suffice for reasonable performance.

In a practical model consisting of more than just the four time points shown in Figure 3, some points will refer to the past and some to the future. One particular point is designated the current time or *Now*. Representing the past and present will allow us to incorporate evidence into the model. By convention, the nodes corresponding to observations are meant to indicate observations *completed* at the associated time point, and nodes corresponding to actions are meant to indicate actions *initiated* at the associated time point. The actions of the robot at past time points and the observations of the robot at past and present time points serve as evidence to provide conditioning events for computing a posterior distribution. For instance, having observed σ at T , denoted $\langle O_R = \sigma, T \rangle$, and initiated α at $T-1$, denoted $\langle A_R = \alpha, T-1 \rangle$, we will want to compute the posterior distribution for S_R at T given the evidence:

$$\Pr(\langle S_R = \omega, T \rangle, \omega \in \Omega_{S_R} | \langle O_R = \sigma, T \rangle, \langle A_R = \alpha, T-1 \rangle).$$

To update the model as time passes, all of the evidence nodes are shifted into the past, discarding the oldest evidence in the process. Figure 4 shows a network with nine time points. The lighter shaded nodes correspond to evidence. As new actions are initiated and observations are made, the appropriate nodes are instantiated as conditioning nodes, and all of the evidence is shifted to the left by one time point.

The darker shaded nodes shown in Figure 4 indicate nodes that are instantiated in the process of evaluating possible sequences of actions. For evaluation purposes, we employ a simple *time-separable* value function. By time separable, we mean that the total value is a (perhaps weighted) sum of the value at the different time points. If V_t is the value function at time t , then the total value, V , is defined as

$$V = \sum_{t \in T} \gamma(t) V_t,$$

where $\gamma : T \rightarrow \{x | 0 \leq x \leq 1\}$ is a decreasing function

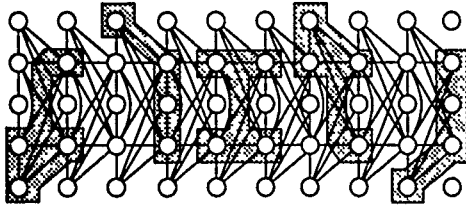


Figure 5: Chordal graph showing example cliques

of time used to discount the impact of future consequences. Since our model assumes a finite \mathcal{T} , we already discount some future consequences by ignoring them altogether; γ just gives us a little more control over the immediate future. For V_t , we use the following function

$$V_t = - \sum_{\substack{\omega_i \in \Omega_{S_T} \\ \omega_j \in \Omega_{S_T}}} \Pr(\langle S_T = \omega_i, t \rangle) \Pr(\langle S_T = \omega_j, t \rangle) \text{Dist}(\omega_i, \omega_j),$$

where $\text{Dist} : \Omega_{S_T} \times \Omega_{S_T} \rightarrow \mathbb{R}$ determines the relative Euclidean distance between pairs of locations. The V_t function reflects how much uncertainty there is in the expected location for the target. For instance, if the distribution for $\langle S_T, t \rangle$ is strongly weighted toward one possible location in Ω_{S_T} , then V_t will be close to zero. The more places the target could be and the further their relative distance, the more negative V_t .

The actions in Ω_{A_R} consist of tracking and localization routines (e.g., move along the wall on your left until you reach a corner). Each action has its own termination criteria (e.g., reaching a corner). We assume that the robot has a set of strategies, \mathcal{S} , consisting of sequences of such actions, where the length of sequences in \mathcal{S} is limited by the number of present and future time points. For the network shown in Figure 4, we have

$$\mathcal{S} \subset \Omega_{A_R} \times \Omega_{A_R} \times \Omega_{A_R} \times \Omega_{A_R}.$$

The expectation is that \mathcal{S} is actually quite small, since we propose evaluating the network $|\mathcal{S}|$ times at every decision point. The strategy with the highest expected value is that strategy, $\varphi = \alpha_0, \alpha_1, \alpha_2, \alpha_3$, for which V is a maximum, conditioning on $\langle A_r = \alpha_0, \text{Now} \rangle$, $\langle A_r = \alpha_1, \text{Now}+1 \rangle$, $\langle A_r = \alpha_2, \text{Now}+2 \rangle$, and $\langle A_r = \alpha_3, \text{Now}+3 \rangle$. The best strategy to pursue is reevaluated every time that an action terminates.

We use Jensen's [1989] variation on Lauritzen and Spiegelhalter's [1988] algorithm to evaluate the decision network. The cost of evaluating a Bayes network using this algorithm is largely determined by the product of the sizes of the sample spaces for the nodes in the largest clique of the chordal graph formed by first moralizing and then triangulating the underlying graph [Jensen, 1989]. Figure 5 shows representative cliques for networks of the sort shown in Figure 4.

For our network model, the computational cost of evaluation is roughly proportional to the product of $|\mathcal{T}|$,

$|\Omega_{S_T}|^2$, and $|\Omega_{S_R}|^2$. We could reduce Ω_{S_R} and Ω_{S_T} to a size that would allow for reasonably quick evaluation by using a very coarse tessellation of the geometric model of the environment. However, this coarse tessellation would provide little useful guidance for tracking. As an alternative to using a coarse tessellation, we are exploring a method for modifying Ω_{S_R} and Ω_{S_T} on the fly by restricting the range of S_R and S_T on the basis of evidence not accounted for in the decision model (e.g., odometry and compass information). It is our expectation that this hybrid model will retain many of the desirable theoretical properties of the temporal Bayes network formulation, while providing practical performance in realistic environments.

Emergent Behavior

Different tracking behaviors will manifest themselves depending on the specifics of the model. For instance, under certain circumstances, the robot will tend to localize itself if it has lost the target. This tendency will be mediated by the preferences specified in the target model. For instance, if the target spends much of its time in offices with southern exposures, the robot will tend to gravitate toward such sunny locations regardless of whether offices with southern exposures are easy to distinguish from one another.

As another example of emergent behavior, if the robot somehow always knows the location of the target relative to itself, then it will tend to localize itself with respect to the map. If there is some special vantage point that would provide a global perspective, then the robot will try to obtain that vantage if it is also a landmark.

The above sort of *emergent* behaviors (i.e., behaviors that are not explicitly programmed into the robot, but that emerge as a consequence of the model) are typical of what is to be expected from systems controlled by decision theoretic criteria. In a decision theoretic approach to control, the designer provides the necessary expectations, actions, and utilities, and the emergent behavior of the resulting system is just that which maximizes utility given the evidence. There is a price to pay for this generally desirable, but more-or-less effortlessly emerging behavior in terms of the computational cost of evaluating the decision model. For practical applications, the system designer has to exercise critical judgement in deciding what to put into the decision model and what to leave out. In some cases, the tradeoffs made by the designer in coping with computational complexity can themselves be quantified by decision theoretic criteria [Dean, 1990].

Note that the finite horizon of the model will tend to restrict behavior. For example, if the robot is at a distinctive place and the target is moving into an area that will confound localization, the robot will tend to remain in place, relying on its ability to track the target. If in some n additional steps the target is likely to be

occluded (e.g., rounding a corner or entering a room), the robot will wait to make its move to follow until the n is small enough to be accounted for in the decision model. The finite horizon also limits the extent to which the robot could be made to perform a systematic search.

Related Work

Bayes networks and influence diagrams are just beginning to see use in robotics and image understanding. Agogino and Ramamurthi [1988] describe the use of influence diagrams for controlling machine tools. In their application, the ability to represent the uncertain effects of vibration on the life of cutting tools in a probabilistic decision model provides an advantage over other expert system technologies. Levitt *et al* [1988] describe an approach to implementing object recognition using Bayes networks that accounts for the cost of sensor movement and inference. Dean *et al* [1990] show how to use Bayes networks and influence diagrams for building maps and reasoning about the costs and benefits of exploration.

With regard to Bayesian decision theory and active sensing, Hager [1988] provides an approach to reducing uncertainty in object recognition. Hager's approach involves the use of search procedures that choose sensor actions that yield the best predicted performance for a given task. Leonard and Durrant-Whyte [1989] provide a method of interpreting sonar data in the presence of specular reflections that serves as the inspiration for our beacon tracking navigation routines.

References

- [Agogino and Ramamurthi, 1988] A. M. Agogino and K. Ramamurthi. Real-time influence diagrams for monitoring and controlling mechanical systems. Technical report, Department of Mechanical Engineering, University of California, Berkeley, 1988.
- [Aloimonos *et al.*, 1987] J. Aloimonos, A. Bandyopadhyay, and I. Weiss. Active vision. In *Proceedings of the First International Conference on Computer Vision*, pages 35–55, 1987.
- [Bajcsy, 1988] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):996–1005, 1988.
- [Ballard, 1989] Dana H. Ballard. Reference frames for animate vision. In *Proceedings IJCAI 11*, pages 1635–1641. IJCAI, 1989.
- [Bülthoff *et al.*, 1989] H. Bülthoff, J. Little, and T. Poggio. A parallel algorithm for real-time computation of optical flow. *Nature*, 337(6207):549–553, February 1989.
- [Camus, 1990] Theodore Camus. Applications of pyramid structures to multiscale optical flow. Technical Report CS-90-09, Brown University Department of Computer Science, 1990.
- [Chapman, 1987] David Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32:333–377, 1987.
- [Dean and Boddy, 1988] Thomas Dean and Mark Boddy. Reasoning about partially ordered events. *Artificial Intelligence*, 36(3):375–399, 1988.
- [Dean and Kanazawa, 1989] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.
- [Dean *et al.*, 1990] Thomas Dean, Kenneth Basye, Robert Chekaluk, Seungseok Hyun, Moises Lejter, and Margaret Randazza. Coping with uncertainty in a control system for navigation and exploration. In *Proceedings AAAI-90*. AAAI, 1990.
- [Dean, 1990] Thomas Dean. Decision-theoretic control of inference for time-critical applications. Technical Report CS-90-44, Brown University Department of Computer Science, 1990.
- [Hager, 1988] Gregory D. Hager. Active reduction of uncertainty in multi-sensor systems. Ph.D. Thesis, University of Pennsylvania, Department of Computer and Information Science, 1988.
- [Howard and Matheson, 1984] Ronald A. Howard and James E. Matheson. Influence diagrams. In Ronald A. Howard and James E. Matheson, editors, *The Principles and Applications of Decision Analysis*. Strategic Decisions Group, Menlo Park, CA 94025, 1984.
- [Jensen, 1989] Finn V. Jensen. Bayesian updating in recursive graphical models by local computations. Technical Report R-89-15, Institute for Electronic Systems, Department of Mathematics and Computer Science, University of Aalborg, 1989.
- [Kanazawa and Dean, 1989] Keiji Kanazawa and Thomas Dean. A model for projection and action. In *Proceedings IJCAI 11*, pages 985–990. IJCAI, 1989.
- [Lauritzen and Spiegelhalter, 1988] Stephen L. Lauritzen and David J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, 50(2):157–194, 1988.
- [Leonard and Durrant-Whyte, 1989] John J. Leonard and Hugh F. Durrant-Whyte. Active sensor control for mobile robotics. Technical Report OUEL-1756/89, Oxford University Robotics Research Group, 1989.
- [Levitt *et al.*, 1987] Tod S. Levitt, Daryl T. Lawton, David M. Chelberg, and Philip C. Nelson. Qualitative landmark-based path planning and following. In *Proceedings AAAI-87*, pages 689–694. AAAI, 1987.
- [Levitt *et al.*, 1988] Tod Levitt, Thomas Binford, Gil Ettinger, and Patrice Gelband. Utility-based control for computer vision. In *Proceedings of the 1988 Workshop on Uncertainty in Artificial Intelligence*, 1988.
- [Pearl, 1988] Judea Pea. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan-Kaufman, Los Altos, California, 1988.

Qualitative Visual Control of a Robot Manipulator *

Jean-Yves Hervé, Peter Cucka and Rajeev Sharma
Computer Vision Laboratory, Center for Automation Research
University of Maryland, College Park, MD 20742

Abstract

Visual input to a robot hand/eye system has traditionally been used only in the calibration step of the expensive process of inverting the robot's kinematic map. We propose a new approach to manipulator positioning tasks in which visual feedback is closely integrated with a robust, qualitative control strategy that does not require preliminary calibration of any of the components of the system. We analyze the topology of the *perceptual kinematic map* between the joint coordinates of the manipulator and a set of image parameters and exploit properties of this map in the design of our control strategy. Thus, relying only on visual knowledge of its configuration, the manipulator is able to successfully maneuver in its work space while avoiding costly and complex calculations. We present preliminary experimental results and outline possible generalizations of this approach.

1 Introduction

As industrial robot manipulators become more precise the inadequacy of their control systems for tasks involving uncertainty becomes more apparent. These systems require complete information about the manipulator's state and the work space geometry—in other words, they require perfect calibration of the robot relative to its environment, a condition that is unlikely to be fulfilled in the real world. Attempts have been made to handle uncertainty by integrating sensors into the control of robot systems. Although most initial attempts have been directed towards mobile robots, sensor-mediated control of manipulators has now become an important research topic, both from theoretical and practical viewpoints.

A robot gripper acting in a 3D work space to grasp a simple, known object requires tactile sensors in order to perform the fine motion of the fingers that will give it a good grip on the object. Correct positioning of the end effector for a grasping task, however, also requires

a distance sensor that is able to provide information on the relationship between the manipulator and its work space. Sonars and ultrasonic sensors are being used for mobile robots, generally to determine maps of the robot's environment ([Crowl85], [Elfes87]). However, they are still too inaccurate to be used in grasping tasks. Laser (short) range finders can provide very reliable information (although black, transparent or very specular objects are known to pose problems) but require complex settings, which restrict their use outside the laboratory ([Klaft89]). The distance sensors which seem the most promising at this time are visual sensors such as CCD cameras. We address the problem of this initial stage of sensor-mediated grasping: positioning of the robot's end effector based on visual information, a problem known as *hand/eye coordination*.

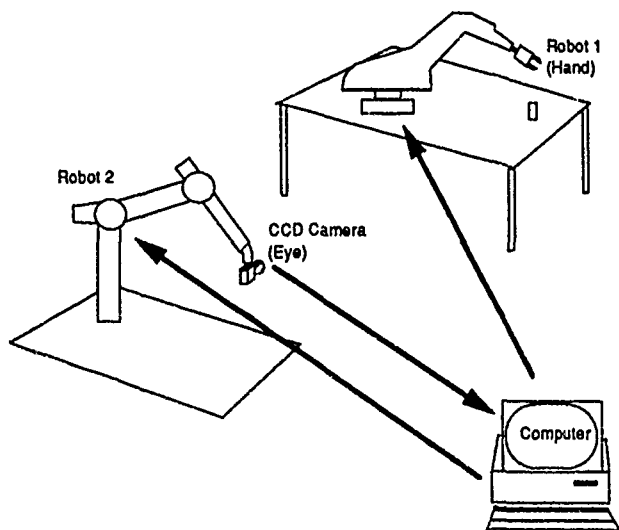


Figure 1: A typical hand/eye system

We have stated that sensors—visual sensors, in our case—are essential to robot systems; however the prevailing opinion (judging from the existing literature, for example [Holle89]) seems to be that they lead to more complications and that they should be used only in a limited way. For example, cameras are typically used only

*The support of the Defense Advanced Research Projects Agency (ARPA Order. No. 6989) and the U.S. Army Engineer Topographic Laboratories under Contract DACA76-89-C-0019 is gratefully acknowledged.

in recognition and detection tasks ([Nitza88], [Sengu87]) or as calibration tools. Calibration techniques for accurate estimation of the configurations of the camera and hand are the subject of much research ([TsaiL89], [Wang87]) but suffer from a number of critical flaws. They require complex calculations, they are sensitive to perturbations of the system, and their use of visual information is minimal. In fact, if inverse kinematics algorithms were sufficiently accurate, visual information would be needed only for the initial calibration of the system, after which the manipulator would proceed to its goal blindly. Unfortunately—as we will show later—this is not possible, and several recalibrations will generally be needed in the realization of a grasping task.

A new approach is emerging in which visual feedback is closely integrated with the control of robot manipulators ([Skaar87], [Weiss87] and [Fedde89]) as well as mobile robots (see [Dickm88] for a good example in which control is simplified through increased visual feedback). Still, a simple grasping action requires good initial calibration of the system and complex inverse kinematics computations.

Because the environments of the robotic systems we have in mind do not offer the laboratory conditions called for by calibration techniques, we propose a control strategy that does not require any calibration of the system. Through a qualitative formulation relying only on visual knowledge of the robot's configuration we can avoid costly and complex calculations, and with the application of a well-constrained and well-understood learning strategy, adaptively interact with the environment as well.

This paper is organized as follows: In Section 2, we motivate our vision-based approach to control. In Section 3, we model the components (robot manipulator and camera) and derive the *forward kinematic map* of the system. In Section 4, we introduce the concepts of *perceptual kinematic map* and *control surface*. In Section 5, the control strategy and algorithms are presented. We conclude with a short description of the current implementation and extensions of this work that we are considering for the near future.

2 Motivation for the Method

Although humans are quite inefficient at estimating distances or angles with the precision needed by a robot manipulator, we are able to perform very complex grasping and fine manipulation tasks. Using mainly visual information—the position, size and orientation of the hand as projected on the retina—we can quickly and accurately position our hands near the objects we want to grasp (Figures 2 and 3).

If an observer knows the sizes of objects in the environment (a very reasonable assumption since the sizes of objects one tries to grasp are virtually always known), the ratio of their sizes on the retina to the size of the observer's hand on the retina provides an estimate of the objects' distances. As the hand moves toward the object, the ratio of sizes gives the observer an indication of the distance remaining and of the *time to collision*, allowing regulation of the hand's velocity and stopping



Figure 2: A grasping task: initial position

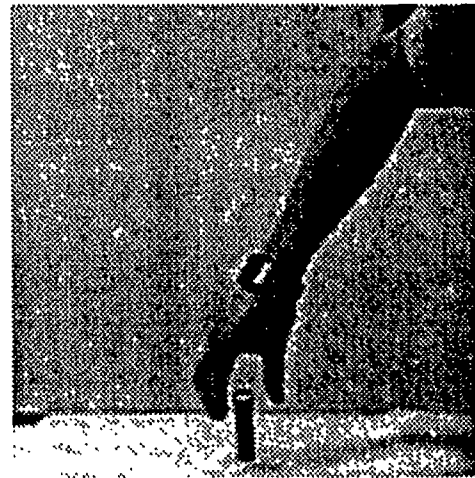


Figure 3: A grasping task: the hand has moved

when the position is "right." This principle should remain valid when the observer is a robot and the hand the end effector of a manipulator.

Since we are not concerned here with fine motion of the hand and fingers, our three-dimensional positioning task corresponds to the two-dimensional problem of positioning image features.

3 Vision-based Kinematics

We first present a quick overview of the *forward kinematics* problem. The controls sent to a robot manipulator affect the configuration of its (step) motors, and thus the position and orientation of its end effector in the work space. The *forward kinematics* of the robot is then modelled by the mapping between the space of all joint (motor) configurations $\mathbf{q} = (q_1, q_2, \dots, q_n)^T$, i.e. the *joint space* \mathcal{J} of the manipulator, and the *task space* \mathcal{K} of possible positions and orientations of the end effector, i.e. the set of all pairs $(\mathbf{p}, \omega) \in \mathbb{R}^3 \times SO(3)$, where $SO(3)$ is the Special Orthogonal Group of three-dimensional rotations.

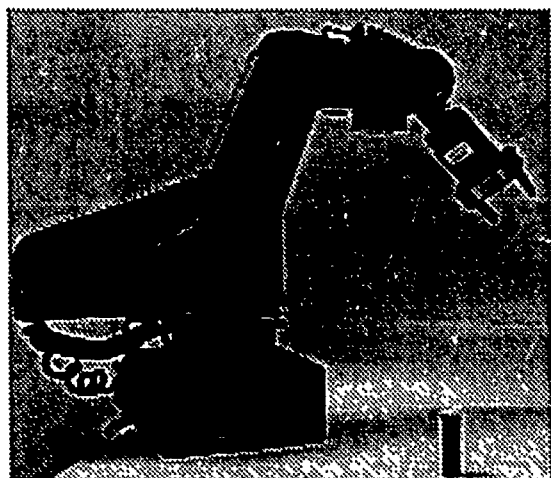


Figure 4: A grasping task: initial position

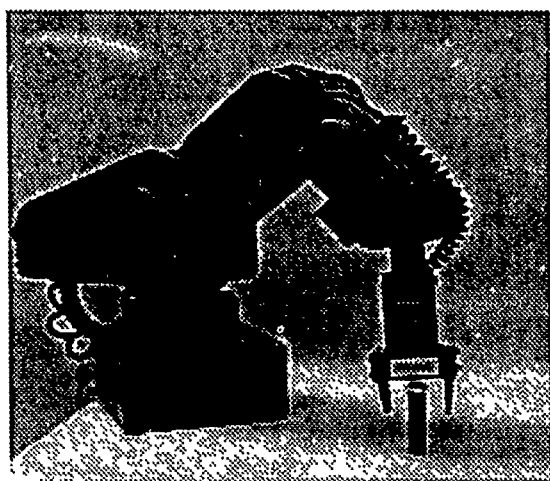


Figure 5: A grasping task: the hand has moved

The (forward) *kinematic map* is then the following mapping:

$$\begin{aligned} \kappa : \mathcal{J} &\longrightarrow \mathcal{K} \\ \mathbf{q} &\longmapsto (\mathbf{p}, \omega). \end{aligned}$$

The first stage in the classical approach to robot control is the establishment of the *kinematic map* for the robot manipulator. This problem has been studied extensively, and numerous models and methods have been proposed for its resolution (good reviews can be found in [Paul81] and [Holle89]). Moving the hand to a given position and orientation involves the solution of an *inverse kinematics* problem. When the Jacobian of the kinematic map is nonzero, the Inverse Function Theorem guarantees locally the existence of a smooth inverse mapping (see Appendix A), but because this mapping is prohibitively difficult to compute, inverse kinematics algorithms generally provide only a fast approximation to the solution. When the Jacobian is zero (i.e. \mathbf{q} is a singular point of the kinematic map), the inversion is even more complex, if not impossible, but it generally

suffices to determine the location of these singularities and to try to avoid them. The general study of singularities of the kinematic map is quite complex, and most articles on the subject concentrate on one particular type of manipulator, e.g. [PaiLe89], [Litvi86] or [Borre86].

3.1 The Denavit-Hartenberg representation

We have chosen to describe the parameters of the robot manipulator using the Denavit-Hartenberg representation ([Denav55] and [Piepe68]), which we now introduce briefly (and denote by D-H from now on). The manipulator is treated as an open linkage, i.e. a sequence of rigid bodies connected by joints that are assumed to be either prismatic (sliding) or revolute (turning). The solids are labeled S_i , in increasing order along the kinematic chain, S_0 being the ground, and S_n the end effector. The joint between S_{i-1} and S_i is denoted J_i . Two reference frames are associated with each solid S_i :

$$R_i = \{O_i, X_i, Y_i, Z_i\}$$

and

$$R'_i = \{O'_i, X'_i, Y'_i, Z'_i\}.$$

The Z_i axis coincides with the axis of joint J_i , while Y_i coincides with the common normal to Z_i and Z_{i+1} (or to OO'). Also Y_i coincides with Y'_i and Z'_i with Z_{i+1} . Whenever the above convention leads to ambiguities in the determination of an axis (such as when Z_i is parallel to Z_{i+1}), a direction is chosen so as to maximize the number of D-H parameters set to zero.

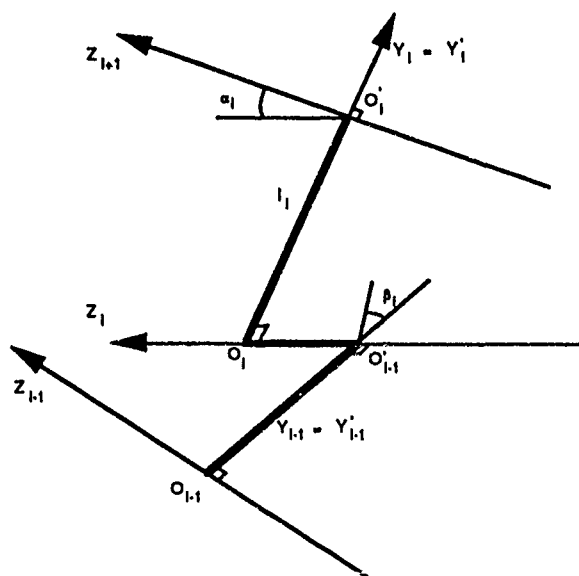


Figure 6: D-H parameters for the general linkage of a robot manipulator

The transformation from one coordinate system to another is described in terms of the parameters $\{\beta_i, d_i, l_i, \alpha_i\}$, where β_i is the angle between Y_{i-1} and Y_i , d_i is the length of $O'_{i-1}O_i$ projected onto Z_i , l_i is

the length of $O_i O'_i$ projected onto Y_i , and α_i is the angle between Z_i and Z_{i+1} (or Z'_i).

The joint parameter q_i can thus be represented as

$$q_i = \sigma_i d_i + \bar{\sigma}_i \beta_i,$$

where σ_i is a Boolean parameter equal to 1 when the joint J_i is prismatic and 0 when J_i is revolute.

This model is very appealing because of its great generality and simplicity. These qualities, however have their drawbacks. The Denavit-Hartenberg representation cannot, for example, handle the inevitable errors in the orientations of the joint axes ([Holle89]). More complex models exist, which require five, six, or even nine parameters ([Vaish87]) for the description of each joint of the robot, but the cost of the *inverse kinematics* computations then becomes too formidable for most (real time) applications.

The homogeneous transformation between the various coordinate systems is given in terms of the D-H parameters as follows: the translation vector, with respect to the reference frame at O_i , is given by

$$O_i O_{i+1} = \begin{bmatrix} 0 \\ l_i \\ d_i \end{bmatrix}_i$$

and the rotation matrices are defined as

$$[R_{i-1',i}] = \begin{bmatrix} \cos \beta_i & -\sin \beta_i & 0 \\ \sin \beta_i & \cos \beta_i & 0 \\ 0 & 0 & 1 \end{bmatrix}_{i-1'}$$

and

$$[R_{i,i'}] = \begin{bmatrix} \cos \alpha_i & 0 & \sin \alpha_i \\ 0 & 1 & 0 \\ -\sin \alpha_i & 0 & \cos \alpha_i \end{bmatrix}_i$$

The coordinates of a point on the end effector (the hand) with respect to the base coordinate system can be obtained using a series of transformations along the kinematic chain S_0, S_1, \dots as is demonstrated in the five degrees-of-freedom (DOF) manipulator that we used in our experiments.

3.2 D-H Parameters for a five-degree-of-freedom robot

A schematic diagram of the 5 DOF robot manipulator is given in Figure 7. Following the convention introduced earlier, the D-H parameters for joints J_1 (the waist) through J_5 (the wrist) are summarized in Table 1.

	i				
d_i	0	0	0	0	d_5
β_i	q_1	q_2	q_3	q_4	q_5
l_i	0	l_2	l_3	0	0
α_i	$\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$	0

Table 1: The D-H parameters for the five DOF robot.

The orientation of the hand with respect to the base is specified as

$$[R]_{0'} = [R_{0',1}][R_{1,2}][R_{2,3}][R_{3,4}][R_{4,5}], \quad (1)$$

where $[R_{i,i+1}] = [R_{i,i'}][R_{i',i+1}]$, and the position of the hand is given in terms of the base reference system as

$$(M)_{0'} = [R_{0',2}] \begin{bmatrix} 0 \\ l_2 \\ 0 \end{bmatrix} + [R_{0',3}] \begin{bmatrix} 0 \\ l_3 \\ 0 \end{bmatrix} + [R_{0',5}] \begin{bmatrix} 0 \\ 0 \\ d_5 \end{bmatrix}. \quad (2)$$

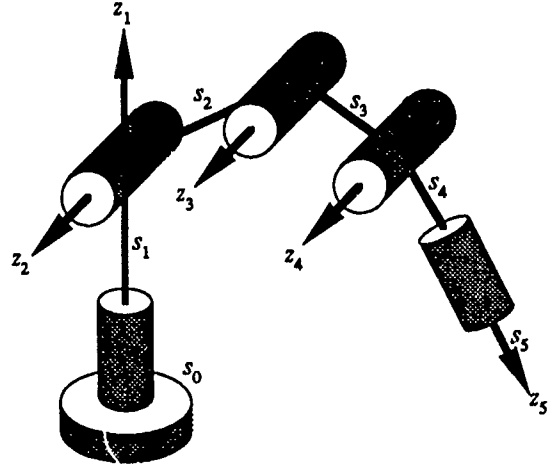


Figure 7: The schematic diagram of a five degree-of-freedom robot manipulator

3.3 Mapping to the Image

Finally, we have to establish the relation between the actions on the robot and what is seen in the image. In order to do this, we adopt a classical pinhole model for the camera.

The 3D world is mapped onto a viewer-based coordinate system defined as follows:

- The origin is the optical center of the camera, O .
- OZ is the optical axis of the camera; it intersects with the image plane orthogonally at o such that $\|\vec{Oo}\| = f$, the focal length of the camera.
- OX and OY are defined so as to be parallel to the axes of the image plane ox and oy (and (X, Y, Z) is direct).

To each of the axes OX, OY, OZ , we associate a unit direction vector, $\mathbf{i}, \mathbf{j}, \mathbf{k}$ respectively. All coordinates will be expressed in the $(O, \mathbf{i}, \mathbf{j}, \mathbf{k})$ system. Let M be a point of the 3D world and $\mathbf{M} = (X, Y, Z)^T$ its coordinate vector. M projects on the image plane as m of coordinate vector $\mathbf{m} = (x, y, f)^T$. Slightly abusing the notation, we will identify the coordinate vector \mathbf{M} (\mathbf{m}) with the position vector \vec{OM} (\vec{Om}). The following relation then holds between \mathbf{M} and \mathbf{m} :

$$\mathbf{m} = \frac{f}{Z} \mathbf{M} = \frac{f}{\mathbf{k} \cdot \mathbf{M}} \mathbf{M}. \quad (3)$$

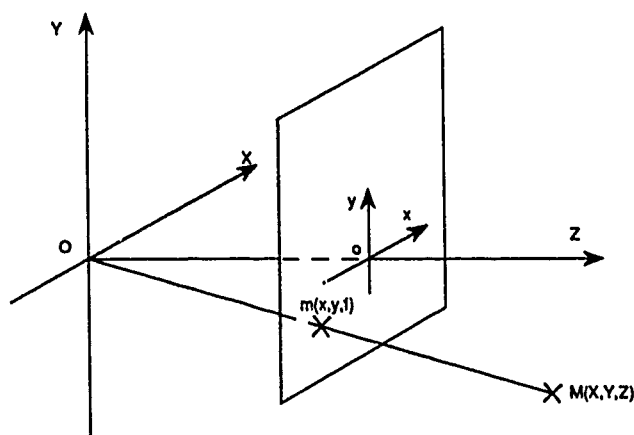


Figure 8: Perspective projection of a world point onto the image plane

In equations (1) and (2) we expressed the kinematic map of the manipulator relative to the coordinate system $R'_0 = \{O'_0, X'_0, Y'_0, Z_0\}$. We can now determine the position and orientation of the hand relative to the camera system O, X, Y, Z by the rotation matrix $[R_{OO'_0}]$

and the translation $O'_0 = \overrightarrow{OO'_0}$. Thus the position in the camera system is

$$(M)_O = (OO'_0)_O + [R_{OO'_0}] \cdot (M)_{O'}, \quad (4)$$

and the orientation is

$$[R]_O = [R_{OO'_0}] \cdot [R]_{O'}. \quad (5)$$

4 The Perceptual Kinematic Map

Having studied the kinematic map $\kappa : \mathcal{J} \rightarrow \mathcal{K}$, we now turn our attention to the movement of the hand and the corresponding changes in the image plane. We consider an array of measurable image parameters $s = (s_1, s_2, \dots, s_n)^T$ as a function of the joint parameters. If \mathcal{S} is the set of such image parameter arrays, then we can consider s as a point of \mathcal{S} and define a mapping $\pi : \mathcal{J} \rightarrow \mathcal{S}$, which we call the *perceptual kinematic map* (PKM). For example, if a robot rotates joint J_i , say its waist, while holding its other joints stationary, we expect that a given point on its hand will trace an arc of an ellipse in the image plane. The coordinates of this point, plotted parametrically against q_i , describe an elliptical cross-section in the x - y plane of the PKM (Figure 9).

For our five-degree-of-freedom robot manipulator we have identified five independent features that can be readily extracted from any image of the robot: the image coordinates of two points on the hand and the area of a rectangle inscribed on the hand. The PKM for this manipulator gives rise in \mathcal{S} to a five-dimensional parametric hypersurface, which we call the *control surface*. (although the term "hypersurface" may be technically a bit inappropriate here, it conveys the image of constrained planning that we want to emphasize.) Through

experiments in which we have tracked the five image features, we have found that their trajectories agree closely with the predicted ones (Figure 10).

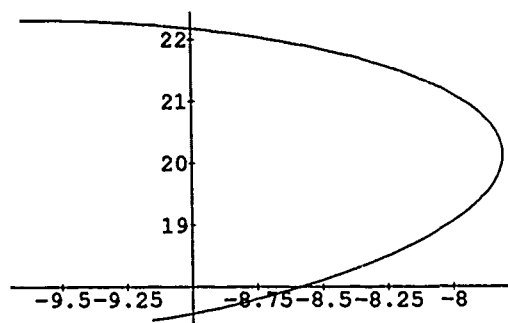


Figure 9: Predicted trajectory of an image point with respect to a single joint

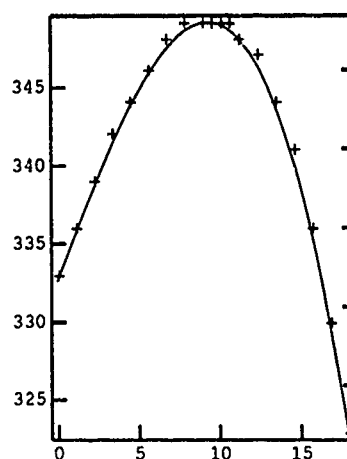


Figure 10: Observed trajectory of one coordinate of an image point

At this stage of our project, we do not deal with the problem of fine motion, so the goal of a grasping task is to superimpose the hand on an object (Figure 5). Both initial and final (expected) positions of the hand correspond to points in \mathcal{S} that lie on the control surface. The grasping task is then reduced to a problem of constrained path planning on the control surface. An obvious generalization of the current system, which we are currently studying, would allow the final position to belong to a goal set rather than be an isolated point (there is clearly more than one correct position for grasping). Also of great practical interest is the determination of this set from the visual input and *a priori* knowledge about the object (its shape, for example).

There is little point in attempting to invert the PKM, since the computations are even more complex than those for the original kinematic map, and small dis-

cretization and detection errors would invalidate the results ([Spets89]). Qualitative decisions, however, such as whether or not the current configuration is close to a singularity of the control surface, are unaffected by small errors. For example, if the trajectory of feature s_i with respect to joint parameter q_i follows an arc of an ellipse then we can expect to encounter singularities at the two points of intersection of the ellipse with its major axis. The results in Figure 10 confirm that the robust detection of these singularities through image measurements is feasible.

The study of the perceptual kinematic map tells us what the control surface can look like and what kinds of singularities can be expected. For example, in Figure 11 one coordinate of an image point is plotted as a function of the motion of two joints. The ability to predict events in the image, and thereby the nature of the control surface, allows us to devise a control strategy that is easy to describe algorithmically. At the same time, we can avoid undue stress on the manipulator and the unnecessary movement that is incurred when it is made to maneuver blindly across a singularity of its control surface.

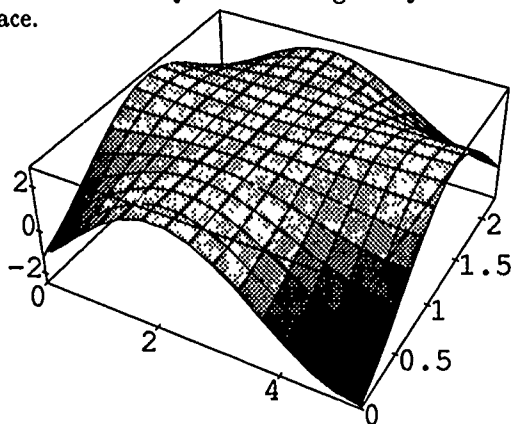


Figure 11: A representative portion of a control surface

To illustrate these concepts, we can consider a two-degree-of-freedom manipulator (and therefore a two dimensional joint space \mathcal{J}^2) such as the one displayed in Figure 12, and a two-dimensional space of image parameters \mathcal{S}^2 consisting of the possible image coordinates of the robot's endpoint. The PKM in this case is fairly simple, reduced to an $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ mapping. From [Whitn55] we know that its *generic* singularities are smooth lines called *fold lines*, and this is quite apparent in Figure 13, which represents a parametric plot of x , y and q_2 as functions of q_1 and q_2 .

5 Hand/Eye Control

The control surface can be divided into regular points, where the control surface can be locally approximated by its tangent hyperplane, facilitating a particularly simple control strategy, and singular points, where the Jacobian of π is zero. Determining whether the present configuration falls at a regular point or a singularity is one of the most important functions of the system. The robot

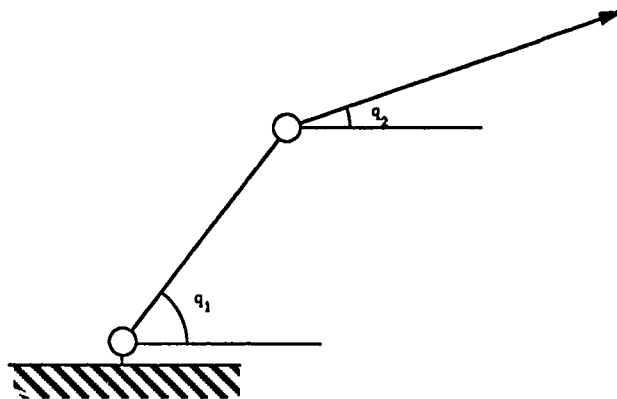


Figure 12: A two-degree-of-freedom robot manipulator

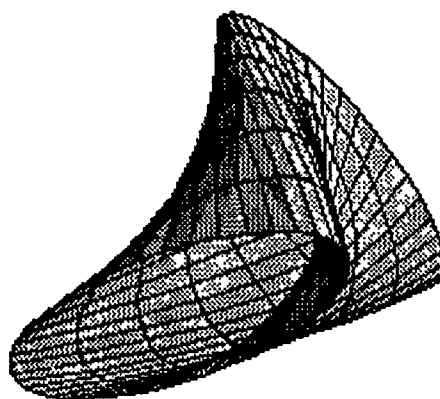


Figure 13: Parametric surface for x, y and q_2 against q_1

attempts to reduce a distance, defined on \mathcal{S} , between its current configuration and the goal point in \mathcal{S} . For each new grasping task the robot first explores its neighborhood on the control surface. If the neighborhood is regular, a simple steepest gradient method can be employed to reduce the distance to the goal. As the robot moves, it updates its estimates of the local directional derivatives of the control surface. In the neighborhood of a singular point, the control surface can have a complex topological structure. If the robot encounters a singularity of the control surface, it must reexplore its neighborhood in order to identify the type of the singularity. Ideally, singularities are identified by zeros of the Jacobian, but since we are dealing with discrete images and manipulator displacements, we can only expect to detect changes in its sign. In its regular control mode, the robot monitors the sign of the Jacobian and can thus detect that it has crossed a singularity.

5.1 Exploration of the control surface

At the beginning of a new grasping task, or when a singularity is encountered, the robot needs to explore its neighborhood, meaning that it computes the Jacobian matrix of the PKM.

The exploration of a configuration's neighborhood proceeds as follows. Let the current configuration be denoted by $\mathbf{q} = (q_1, q_2, q_3, q_4, q_5)^T$, and let the corresponding point on the control surface be denoted by $\mathbf{s} = (s_1, s_2, s_3, s_4, s_5)^T$. In the neighborhood of \mathbf{q} we make a series of measurements $\mathbf{s}(\mathbf{q} + \Delta\mathbf{q})$, where $\Delta\mathbf{q}$ is a discrete displacement in the joint space: $\Delta\mathbf{q} = ((i_1\delta q_1, i_2\delta q_2, \dots, i_5\delta q_5)^T)$, with each $i_j = -k, \dots, k$ (in practice, $k = 1$ or 2), and the δq_j are fixed steps for the joint variables. As an example, we show for $k = 1$ how these measurements are made in the neighborhood of a regular point (Figure 14) and in the neighborhood of a singular point (Figure 15).

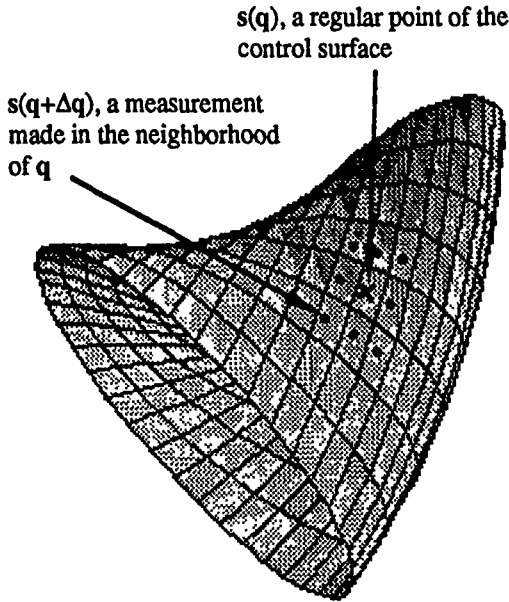


Figure 14: Exploration of the control surface in the neighborhood of a regular point

The list of possible types of singularities of a generic $\mathbb{R}^5 \rightarrow \mathbb{R}^5$ mapping is well known ([Golub73]). Singularities of the control surface will typically be folds, which are degenerate along one direction of the control surface and locally look like five-dimensional cylinders. The direction of a fold, which is defined by the kernel of the Jacobian matrix, can be directly determined from the measurements made, and it qualitatively identifies the singularity. After deciding on which side of the singularity the goal lies, the robot resumes its course, returning to the regular control mode.

5.2 Regular control

As long as the robot does not cross any singularity, it must update its estimate of the tangent hyperplane using only its measurements of the directional derivatives along \mathbf{u} , the direction of displacement. This is realized by a Kalman filter defined as follows:

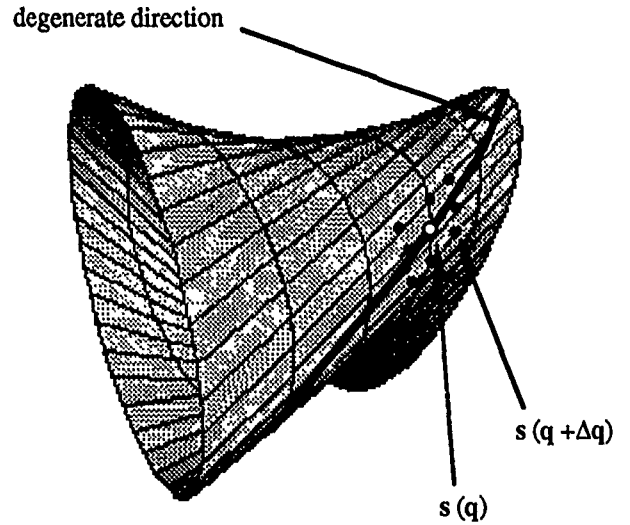


Figure 15: Exploration of the control surface in the neighborhood of a singular point

- The state vector at time k , $\mathbf{x}(k)$, is a 25×1 column vector composed of the elements of the Jacobian matrix $L(k)$:

$$\mathbf{x}(k) = \left[\left(\frac{\partial s_1}{\partial \mathbf{q}} \right), \left(\frac{\partial s_2}{\partial \mathbf{q}} \right), \left(\frac{\partial s_3}{\partial \mathbf{q}} \right), \left(\frac{\partial s_4}{\partial \mathbf{q}} \right), \left(\frac{\partial s_5}{\partial \mathbf{q}} \right) \right]^T \quad (6)$$

It verifies the state equation

$$\mathbf{x}(k+1) = \mathbf{A}(k)\mathbf{x}(k) + \mathbf{v}(k) \quad (7)$$

where $\mathbf{A}(k)$ is the 25×25 state matrix and $\mathbf{v}(k)$ is the 25×1 disturbance vector.

Its initial mean and covariance are known

$$E[\mathbf{x}(k_0)] = \mathbf{x}_0 \quad \text{and} \quad \text{Cov}[\mathbf{x}(k_0)] = \mathbf{p}_0.$$

- The state vector cannot be directly determined at each time $t = kT$, but the derivatives of π along the direction of displacement \mathbf{u} can be measured (see Appendix B for details and a sketch of the algorithms involved). They form the 5×1 measurement vector

$$\mathbf{y}(k) = \left[\left(\frac{\partial s_1}{\partial \mathbf{u}} \right), \left(\frac{\partial s_2}{\partial \mathbf{u}} \right), \left(\frac{\partial s_3}{\partial \mathbf{u}} \right), \left(\frac{\partial s_4}{\partial \mathbf{u}} \right), \left(\frac{\partial s_5}{\partial \mathbf{u}} \right) \right]^T \quad (8)$$

The measurement vector can be expressed as a function of the state vector (see Appendix B for the complete equation)

$$\mathbf{y}(k) = \mathbf{C}(k)\mathbf{x}(k) + \mathbf{w}(k), \quad (9)$$

where $\mathbf{C}(k)$ is the 5×25 measurement matrix and $\mathbf{w}(k)$ is the 5×1 measurement error.

- The following assumptions are made about the noise in the system:

Both $v(k)$ and $w(k)$ are white Gaussian noise processes:

$$E[v(k)] = 0 \quad \text{and} \quad E[v(k)v'(l)] = Q(k)\delta_{kl}$$

$$E[w(k)] = 0 \quad \text{and} \quad E[w(k)w'(l)] = R(k)\delta_{kl}.$$

They are uncorrelated with the initial state and with each other:

$$E[x(k_0)v(k)] = 0$$

$$E[x(k_0)w(k)] = 0$$

$$E[v(k)w(l)] = 0.$$

We seek, on the basis of the measurements and of an initial value of x_0 , an unbiased linear estimate of $x(k)$, which we will call $\hat{x}(k)$, and which minimizes the variance estimate

$$p(k) = E[(x(k) - \hat{x}(k))(x(k) - \hat{x}(k))'].$$

The solution to this problem is given by the Kalman-Bucy algorithm ([Bramm89]) for which a block diagram is given in Figure 16.

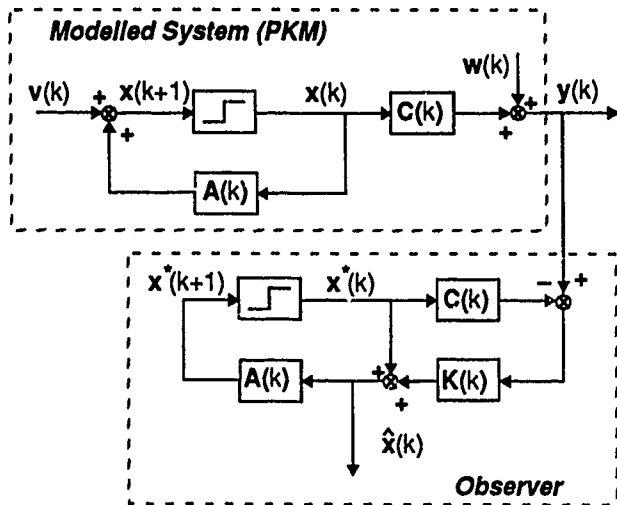


Figure 16: Block diagram of the Kalman-Bucy filter

- Prediction phase:

$$x^*(k+1) = A(k)\hat{x}(k)$$

$$p^*(k+1) = A(k)\tilde{P}^*(k)A'(k)$$

- Update phase:

$$\hat{x}(k+1) = x^*(k+1) + K(k)\{y(k) - C(k)x^*(k)\}$$

$$K(k) = p^*(k)C'(k)\{C(k)p^*(k)C'(k) + R(k)\}^{-1} \quad (10)$$

$$\tilde{P}^*(k) = p^*(k) - K(k)C(k)p^*(k)$$

- Initial conditions:

$$x^*(k_0) = x_0$$

$$p^*(k_0) = p_0.$$

It should be pointed out that, although the state vector is composed of 25 rows, our measurement vector is only 5×1 , which means the square matrix we have to invert in (10) at each iteration is only 5×5 .

Some comments have to be made on the choice of $A(k)$ in (7). The determination of a "good" $A(k)$ would necessitate an exact (read quantitative) knowledge of mapping π which, as we previously explained, is exactly what we want to avoid. What we would need is a *qualitative Kalman Filter*, which would allow us to exploit our knowledge of the topology of the control surface (for example, we know that (s_1, s_2) describes an arc of an ellipse when one joint is moved, the others being fixed). Unfortunately, such a mathematical tool does not exist yet. At this point in our project, we use $A(k) = I$ (the 25×25 identity matrix). This choice may seem surprising, as it provides minimum information to the Kalman filter, indicating only that if the displacement steps are small enough, the partial derivatives on the PKM vary slowly. However, experimental data (Figures 17, 18 and 19) show that this hypothesis is in fact quite close to reality.

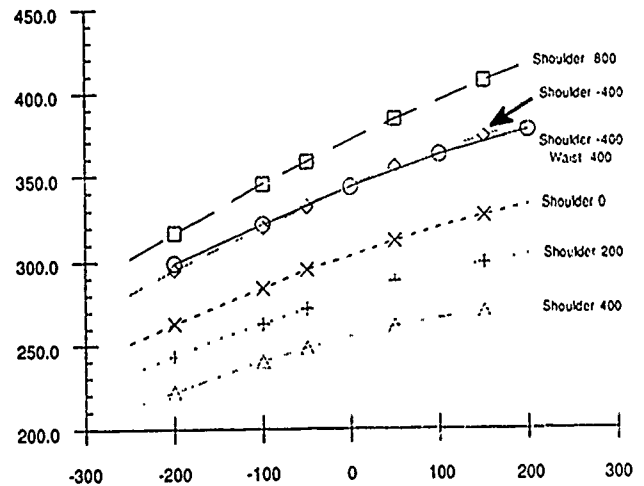


Figure 17: Experimental result: x coordinate vs wrist

5.3 Qualitative learning of the environment

In its initial ignorance of their locations, the robot can be expected to encounter one or more singularities on its path to the goal, but by monitoring changes in the signs of curvatures along the path, the robot can recognize and record these singularities. In doing so over the course of a number of grasping tasks, the robot learns the topology of the control surface. Indeed, knowledge of the type and location of its singularities uniquely defines a control surface in a qualitative way. This means that when all the singularities of the control surface are localized and identified, the robot would have achieved qualitative control over its environment.

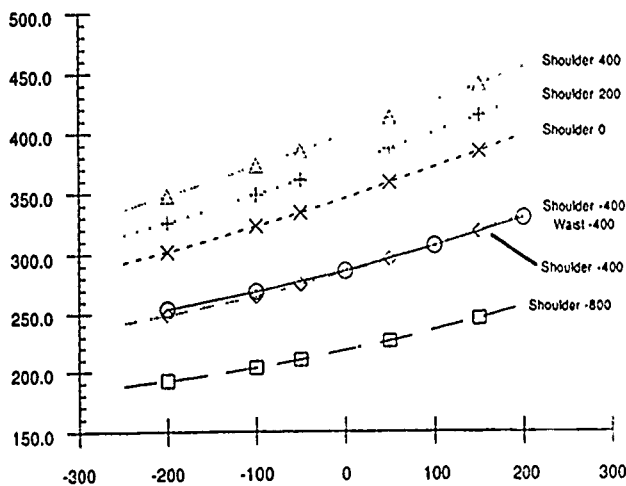


Figure 18: Experimental result: y coordinate vs wrist

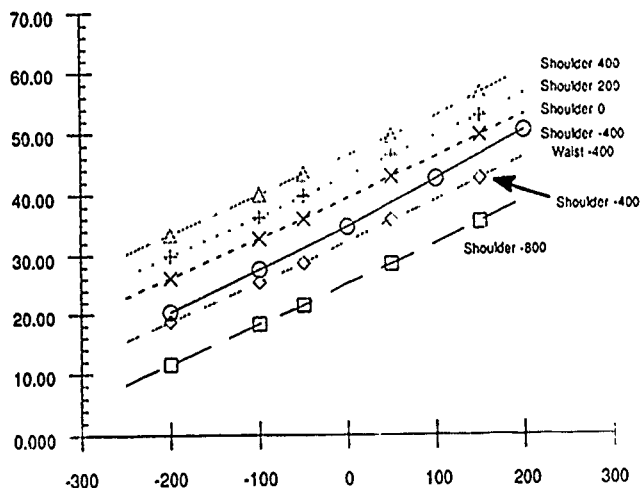


Figure 19: Experimental result: orientation vs wrist

6 Implementation

Our experimental system consists of an observer, a robot manipulator and an object to be grasped (Figure 4). Both the camera and the manipulator are uncalibrated—their relative position and orientation are unknown.

The observer is a CCD camera (Sony, Model XC-38, genlock unit CBK-38 GL, lens system Switar, focal length 10mm) mounted on a six-degree-of-freedom American Robot robot arm. The gripper is a five-degree-of-freedom Mitsubishi MoveMaster II robot arm. Image processing and robot control are performed by a Macintosh IIcx.

At this stage of the project, the image processing is purposely kept simple. A white rectangle is inscribed on the hand in order to facilitate the tracking of the image parameters: its position and area in the image giving the five parameters needed for the control (Figures 20 and 21).



Figure 20: The raw image of the hand.

7 Conclusion and further research

We have presented the concepts of the *perceptual kinematic map* and *control surface* of a hand/eye system and shown that complex tasks can be simply modelled and solved in terms of these concepts. The close integration of the visual information to the decision (control) process and our relying on qualitative (topological) information rather than on the exact values of the maps allow us to achieve robust visual control of a robot manipulator without preliminary calibration of any of the components of the system. The principles of our approach are general, and the algorithm and system presented here are only one of the many possible ways to implement and test them. Although they proved to be successful, they are in no way indicative of what future systems based on the same principles may look like.

A number of questions still remain to be addressed, among them: how can the robot generalize its knowledge of the control surface for one fixed camera position to another, and how can it best exploit a mobile camera? The concept of an observer acting on its environment, or at least on the parameters that control the images it receives, was only recently introduced ([Bajcs86]). More recently, [Aloim87] showed that most "shape from x " problems can be made simpler and their solutions more robust when an active observer is involved. The determination of a "best" activity was shown to be possible, but the resulting optimization problem is quite involved ([Herv90d]). In the context of our qualitative approach to computer vision, the active observer is concerned not with the effect of its actions on the solution to a "shape

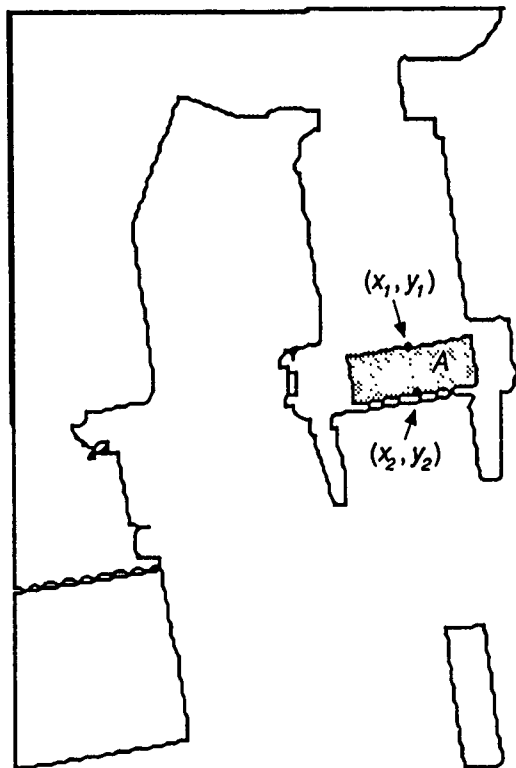


Figure 21: The observed image parameters.

from x problem but rather with their effect on the topological structure of the control surface that is, on the singularities of the PKM (or equivalently, on the zero set of the PKM's Jacobian). The effects of classes of perturbations on the solutions of an equation have been thoroughly studied, at least in the case of low (≤ 5) codimension (see [Golub88] and [Golub85]). The inverse problem, however, of determining a motion (perturbation) that achieves a particular effect on the control surface remains complex.

Given the qualitative nature of the control strategy, we are also interested in determining if the robot can achieve the fine control necessary for the final stages of a grasping task. We see two mutually compatible solutions to this problem. The first solution is to increase the resolution of the image so that the same control techniques can be applied to each of the fingers. We consider this action to be a subcase of the active observer, since increased resolution can be obtained by bringing the camera closer to the scene. The second solution is to add additional (tactile) sensors to the system.

Another direction of improvement of the current system was discussed in Section 5.2 and concerns the transition matrix of the Kalman filter. In order to reduce the burden on the filter's mechanism, we need to define a better $A(k)$ as compared to the identity matrix we are currently using. We are studying the idea of a Kalman filter that is able to handle qualitative information, for example the sign of the surface's curvature.

As pointed out earlier in this paper, we are currently working on an extension of the control strategy to the

case of a final goal set rather than an isolated point, and to the determination of this goal set from visual and *a priori* clues on the object to be grasped.

Finally, we intend to realize very soon a real-time implementation of our system (within the limits allowed by our current hardware).

Acknowledgments

We would like to thank Dr. John Aloimonos for his advice and constructive criticism, and the CVL Qualitative Vision group for endless discussions.

A Local inversion of the PKM

We present here a justification for our linear approximation of the PKM and, more generally, our choice of control strategy in Section 5. Although we have tried to give only an overview of the problem (more rigorous and complete treatments can be found in [Arnol73], [Gibso79] and [Chill76]), the reader's understanding and intuition may benefit from the following exposition. We begin with an informal description of the result:

Inversion Problem

We prove that, whenever the Jacobian of the PKM is nonzero, successful control can be achieved by following the inverse image of a path in the image parameter space S . Furthermore, the linear approximation of the PKM is locally acceptable, so that the manipulator's control is uniquely determined by simple local inversion of a linear mapping.

Let E and F be Banach spaces, U be an open subset of E and $f : U \rightarrow F$ be a mapping from U to F . (We will reserve the use of the word *function* for the case of a mapping $U \rightarrow \mathbb{R}$.)

Definition¹ A mapping f is called a *homeomorphism* if

- f is a bijection,
- f is continuous, and
- its inverse mapping f^{-1} is continuous.

Definition A mapping f is *differentiable* at $x_0 \in U$ if there exists a linear map $Df(x_0) : E \rightarrow F$ such that

$$\forall h \in E, f(x_0 + h) = f(x_0) + Df(x_0) \cdot h + \|h\|_E \nu(h),$$

where $\nu(h) \in F$ and $\|\nu(h)\|_F \rightarrow 0$ as $\|h\|_E \rightarrow 0$. We call $Df(x_0)$ the *derivative* of f at x_0 .

When $E = \mathbb{R}^n$ and $F = \mathbb{R}^m$, which is the case of the PKM π , f can be written as an n -tuple of functions $E \rightarrow \mathbb{R}$: $f(x) = (f_1(x), f_2(x), \dots, f_m(x))$, and $Df(x_0)^T$ is

¹The structures of Banach spaces are not necessary for the definition of a homeomorphism. Structures of Hausdorff topological spaces for the domain and range of f suffice.

represented in the standard bases of E and F by the *Jacobian matrix* of f :

$$\frac{\partial(f_1, f_2, \dots, f_m)}{\partial(x_1, x_2, \dots, x_n)} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \dots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

If $n = m$ then the determinant of the Jacobian matrix at \mathbf{x}_0 exists. We call it the *Jacobian* of f at \mathbf{x}_0 and denote it by $Jf(\mathbf{x}_0)$, or simply J when the context is unambiguous.

Definition A mapping f is called a *diffeomorphism* (C^r *diffeomorphism*) if

- f is a bijection,
- f is differentiable (C^r differentiable), and
- its inverse mapping f^{-1} is differentiable (C^r differentiable).

Inverse Function Theorem²

Let $f : U \rightarrow F$ be a C^r map ($1 \leq r \leq \infty$ or $r = \omega$). Suppose \mathbf{x}_0 is a point in U for which $Df(\mathbf{x}_0) : E \rightarrow F$ is an isomorphism. Then there is a neighborhood $V \subset U$ of \mathbf{x}_0 such that $f|_V : V \rightarrow f(V)$, the restriction of f to V , is a C^r diffeomorphism.

The intuitive meaning of this theorem, for $E = \mathbb{R}^n$ and $F = \mathbb{R}^m$, is that we can change coordinates locally in E to convert f into its own linear approximation at \mathbf{x}_0 .

If $E = \mathbb{R}^n$ and $F = \mathbb{R}^n$, then $Df(\mathbf{x}_0)$ is bijective if and only if $Jf(\mathbf{x}_0) \neq 0$.

In the case of the PKM $\pi : \mathcal{J} \rightarrow \mathcal{S}$, where $\mathcal{J} \subset \mathbb{R}^5$ and $\mathcal{S} \subset \mathbb{R}^5$, we would like to determine a sequence of joint coordinates, i.e. a path, that achieves a given effect on the image parameter space \mathcal{S} . The Inverse Function Theorem tells us that whenever the Jacobian $J\pi(\mathbf{q})$ is nonzero, the map can be inverted on a neighborhood of \mathbf{q} . We are interested in the images of paths in \mathcal{J} , i.e. the effect of the controls we can apply, and the images through π^{-1} of paths in \mathcal{S} (that can produce the effects we want to achieve).

Definition A *smooth path* in U based at \mathbf{q} is a smooth map $p : I \rightarrow U$, where I is some open interval of \mathbb{R} (a, b), with $a < 0 < b$, $U \subset E$ and $p(0) = \mathbf{q}$.

If a map $f : U \rightarrow F$ is smooth then

$$\begin{aligned} D(f \circ p)(0) &= Df(p(0)) \circ Dp(0) \\ &= Df(\mathbf{q}) \cdot p'(0). \end{aligned}$$

This means that the derivative of f at \mathbf{q} takes the tangent to the path p to the tangent into the path $f \circ p$ at $f(\mathbf{q})$ (Figure 22). A consequence of this result is that

²This theorem is equivalent to the Implicit Function Theorem, which is more frequently presented in textbooks.

the derivative $Df(\mathbf{q})$ is completely characterized by the effect of f on smooth paths in U based at \mathbf{q} . This intuitive observation will now be formalized to help define the *tangent map* of f , the last step in the solution of our problem.

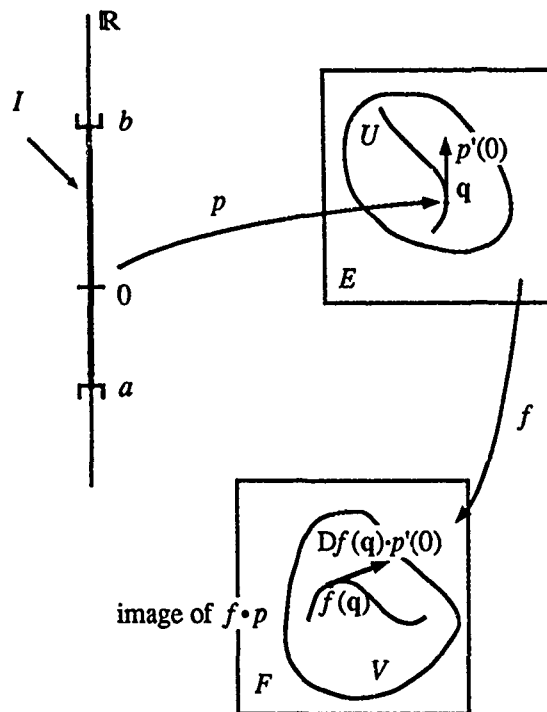


Figure 22: Tangent to a smooth path (redrawn from [Chill76]).

Definition Two smooth paths p_1 and p_2 based at \mathbf{q} are *tangent* (or *contact equivalent*) if

$$\lim_{t \rightarrow 0} \frac{1}{t} (p_1(t) - p_2(t)) = 0 \quad \text{in } E.$$

Tangency is obviously an equivalence relation on the set of all paths in U based at \mathbf{q} . We denote the tangency class of a path p by $[p]$ and call $T_{\mathbf{q}}U$, the set of tangency classes in U based at \mathbf{q} , the *tangent space* to U at the point \mathbf{q} . This space is isomorphic to E , which means that we can transform the linear map $Df(\mathbf{q}) : E \rightarrow F$ into the linear map

$$T_{\mathbf{q}}f : T_{\mathbf{q}}U \rightarrow T_{f(\mathbf{q})}V,$$

which takes each $[p]$ in $T_{\mathbf{q}}U$ to $[f \circ p]$ in $T_{f(\mathbf{q})}V$

We have explained that the grasping task is a problem of trajectory planning on the control surface; in order to solve it, we need to determine the joint control such that the point of \mathcal{S} defined by the hand parameters follows a given path. Let \mathbf{q} be the current configuration of the joints (motors) of the robot and \mathbf{s} the corresponding point in the image space \mathcal{S} . If $J\pi(\mathbf{q}) \neq 0$ (away from

singularities), π diffeomorphically maps a neighborhood U of \mathbf{q} onto a neighborhood V of \mathbf{s} . The tangent map at \mathbf{q} $T\pi(\mathbf{q})$ is thus a local diffeomorphism $T_{\mathbf{q}}U \rightarrow T_{\mathbf{s}}V$.

Let g be a smooth path based in V at \mathbf{s} we want the robot to follow. Then there exists a unique $[p] \in T_{\mathbf{q}}U$ such that its image through $T\pi(\mathbf{q})$ is $[g]$. Furthermore, we know by the Inverse Function Theorem that we can replace π on U by its linear approximation, so that the direction of the control p achieving $[g]$ at \mathbf{q} is given (with considerable abuse of notation) by

$$\frac{\partial(\pi_1, \pi_2, \dots, \pi_5)}{\partial(q_1, q_2, \dots, q_5)} \cdot [p] = [g]$$

which uniquely defines $[p]$ since, by hypothesis, the Jacobian matrix is invertible.

Appendix B will show that the solution of this linear equation is in fact not necessary for the determination of the inverse mapping $\mathbf{q} = \pi^{-1}(\mathbf{s})$.

B Cost function, steepest descent, and numerical differentiation

To reiterate, our goal is to reach a final position $\mathbf{s}_f \in \mathcal{S}$ from a starting position $\mathbf{s}_0 \in \mathcal{S}$, that is, to define a smooth path lying on the control surface that joins these two points. In Appendix A we showed that, given such a path, the PKM can be locally inverted to determine an adequate control sequence for the joint parameters.

A simple way to obtain a path between \mathbf{s}_0 and \mathbf{s}_f is to define a cost function $\mathcal{C}(\mathbf{s}) = d(\mathbf{s}, \mathbf{s}_f)$ on \mathcal{S} that measures some "distance" to the goal \mathbf{s}_f . For example, we can choose the mean squared error $\mathcal{C}(\mathbf{s}) = \frac{1}{2} \sum_{i=1}^5 (\mathbf{s}_i - \mathbf{s}_{f,i})^2$. This choice of \mathcal{C} helps define the "global" cost function $\mathcal{G} = \mathcal{C} \circ \pi$ which estimates distances on the joint space \mathcal{J} . From Appendix A, we know we can represent the derivative of \mathcal{G} at \mathbf{q} , $D\mathcal{G}(\mathbf{q})$, by the Laplacian (row) matrix $\frac{\partial \mathcal{G}}{\partial \mathbf{q}}$ and obtain the following matrix equation in a neighborhood U of \mathbf{q} :

$$\mathcal{G}(\mathbf{q} + \epsilon \mathbf{u}) = \mathcal{G}(\mathbf{q}) + \left(\frac{\partial \mathcal{G}}{\partial \mathbf{q}} \right) \cdot \epsilon \mathbf{u} + |\epsilon| \nu(\epsilon \mathbf{u}), \quad (11)$$

where $\nu(\epsilon \mathbf{u}) \in \mathcal{S}$ and $\|\nu(h)\|_{\mathcal{S}} \rightarrow 0$ as $\|h\|_{\mathcal{E}} \rightarrow 0$.

Let \mathbf{g} be a unit vector in the direction of $\left(\frac{\partial \mathcal{G}}{\partial \mathbf{q}} \right)$:

$$\left(\frac{\partial \mathcal{G}}{\partial \mathbf{q}} \right) = \mathbf{g}^T \cdot \left[\sum_{i=1}^5 \left(\frac{\partial \mathcal{G}}{\partial q_i} \right)^2 \right]^{\frac{1}{2}}$$

If we consider only the linear term in (11) (a valid approximation when $J\pi(\mathbf{q}) \neq 0$), we can rewrite this equation as

$$\mathcal{G}(\mathbf{q} + \epsilon \mathbf{u}) = \mathcal{G}(\mathbf{q}) + \epsilon \left[\sum_{i=1}^5 \left(\frac{\partial \mathcal{G}}{\partial q_i} \right)^2 \right]^{\frac{1}{2}} \mathbf{g}^T \cdot \mathbf{u}. \quad (12)$$

It follows that, in order to maximize the change in \mathcal{G} , we have to choose $\mathbf{u} = -\mathbf{g}$, so that the iterative step followed is

$$\mathbf{q}(k+1) = \mathbf{q}(k) - \gamma \left(\frac{\partial \mathcal{G}}{\partial \mathbf{q}} \right)^T,$$

where γ is an arbitrary gain. We can now replace \mathcal{G} by $\mathcal{C} \circ \pi$:

$$\frac{\partial \mathcal{G}}{\partial \mathbf{q}} = \frac{\partial}{\partial \mathbf{q}} (\mathcal{C} \circ \pi) = \frac{\partial \mathcal{C}}{\partial \mathbf{s}} \cdot \frac{\partial \pi}{\partial \mathbf{q}}$$

so that the final expression of the iteration step is

$$\mathbf{q}(k+1) = \mathbf{q}(k) - \gamma \left(\frac{\partial \pi}{\partial \mathbf{q}} \right)^T \cdot (\mathbf{s} - \mathbf{s}_f).$$

The last problem remaining concerns the numerical computation of the partial derivatives. As explained in Section 4, although we need an estimate of the Jacobian matrix $\left(\frac{\partial \pi}{\partial \mathbf{q}} \right)$ that defines the state vector of our Kalman-Bucy filter, we can only obtain the directional derivative of the PKM along the direction of motion, say \mathbf{u} : $\left(\frac{\partial \mathcal{G}}{\partial \mathbf{u}} \right) = \left(\frac{\partial \mathcal{G}}{\partial \mathbf{q}} \right) \cdot \mathbf{u}$ (the measurement vector of the filter).

We can now express the measurement matrix $\mathbf{C}(k)$ of (9) in terms of the direction of displacement

$$\begin{aligned} \mathbf{y}(k) &= \begin{bmatrix} \mathbf{u}^T & & \dots & & 0^T \\ & \mathbf{u}^T & & & \\ \vdots & & \mathbf{u}^T & & \vdots \\ 0^T & & \dots & \mathbf{u}^T & \end{bmatrix} \cdot \mathbf{x}(k) \\ &= \mathbf{C}(k) \cdot \mathbf{x}(k). \end{aligned}$$

Since the PKM is well-behaved, an accurate estimate of the directional derivative $\mathbf{y}(k)$ is given by the derivative of the quadratic interpolant of the measurements $\mathbf{s}(k), \mathbf{s}(k-1), \mathbf{s}(k-2)$.

References

- [Ahmad88] S. Ahmad and S.F. Luo, "Analysis of Kinematic Singularities for Robot Manipulators in Cartesian Coordinate Parameters," in *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, Philadelphia, Pennsylvania, 1988, pp. 840-845.
- [Aloim87] J. Aloimonos, I. Weiss and A. Bandyopadhyay, "Active Vision," in *Proceedings of the 1987 DARPA Image Understanding Workshop*, Los Angeles, California, 1987, pp. 167-174.
- [Arnol73] V.I. Arnol'd, *Ordinary Differential Equations*, MIT Press, Cambridge, Massachusetts, 1973.

- [Arnol68] V.I. Arnol'd, "Singularities of Smooth Mappings," *Russian Mathematical Surveys*, 23: 1-43, 1968.
- [Bajcs86] R. Bajcsy, "Passive Perception vs. Active Perception," in *Proceedings of the IEEE Workshop on Computer Vision*, Ann Arbor, Michigan, 1986.
- [Borre86] P. Borrel and A. Liegeois, "A study of multiple manipulator inverse kinematic solutions with applications to trajectory planning and workspace determination," in *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, California, 1986, pp. 1180-1185.
- [Bramm89] K. Brammer and G. Siffing, *Kalman-Bucy Filters*, Artech House, Norwood, MA, 1989.
- [Brost88] R.C. Brost, "Automatic Grasp Planning in Presence of Uncertainty," *International Journal of Robotics Research*, 7(1), 1988.
- [Chill76] D.R.J. Chillingworth, *Differential Topology with a View to Applications*, Pitman, London, 1976.
- [Crowl85] J.L. Crowley, "Dynamic World Modelling for an Intelligent Robot Using a Rotating Ultra-Sonic Ranging Device," in *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, St. Louis, Missouri, 1985, pp. 128-135.
- [Denav55] J. Denavit and R.S. Hartenberg, "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," *ASME Journal of Applied Mechanics*, June 1955, pp. 215-221.
- [Dickm88] E. D. Dickmanns and V. Graefe, "Dynamic Monocular Machine Vision," in *Machine Vision and Applications*, Springer-Verlag, New York, 1988, pp. 223-240.
- [Elfes87] A. Elfes, "Sonar-based Real World Mapping and Navigation," *IEEE Journal of Robotics and Automation*, 3(3), 1987.
- [Evere87] L. Everett, M. Driels and B. Mooring, "Kinematic Modeling for Robot Calibration," in *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, Raleigh, North Carolina, 1987, pp. 183-190.
- [Evere88] L. Everett and T.W. Hsu, "The Theory of Kinematic Parameter Identification for Industrial Robots," *Journal Dyn. Systems, Measurement, and Control*, 110(1):96-100, 1988.
- [Evere89] L. Everett, "Forward Calibration of Closed-Loop Jointed Manipulators," *International Journal of Robotics Research*, 8(4):85-91, 1989.
- [Fedde89] J.T. Feddema and O.R. Mitchell, "Vision-Guided Servoing with Feature-Based Trajectory Generation," *IEEE Transactions on Robotics and Automation*, 5(5):691-700, 1989.
- [Gibso79] C.G. Gibson, *Singular Points of Smooth Mappings*, Pitman, London, 1979.
- [Golub88] M. Golubitsky, I. Stewart and D.G. Schaeffer, *Singularities and Groups in Bifurcation Theory*, vol. II, Springer-Verlag, New York, 1988.
- [Golub85] M. Golubitsky and D.G. Schaeffer, *Singularities and Groups in Bifurcation Theory*, vol. I, Springer-Verlag, New York, 1985.
- [Golub73] M. Golubitsky and V. Guillemin, *Stable Mappings and their Singularities*, Springer-Verlag, New York, 1973.
- [Herv90d] J-Y Hervé and J. Aloimonos, "Shading into Texture and Texture into Shading: an Active Approach," in *Proceedings of the First European Conference on Computer Vision*, Nice, France, 1990.
- [Holle89] J.M. Hollerbach, "A Survey of Kinematic Calibration," in *The Robotics Review* 1, MIT Press, Cambridge, Mass., 1989, pp. 207-242.
- [Klaft89] R.D. Klafter, T.A. Chmielewski and M. Negin, *Robotic Engineering. An Integrated Approach*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.
- [Kumar81] A. Kumar and K.J. Waldron, "Numerical Plotting of Positioning Accuracy of Manipulators," *Mechanism Mach. Theory*, 16(4):361-368, 1981.
- [LaiYa86] Z.C. Lai and D.C. H. Yang, "A New Method for the Singularity Analysis of Simple Six-Link Manipulators," *International Journal of Robotics Research*, 5(2):66-74, 1986.
- [Litvi86] F.L. Litvin, Z. Yi, V. Parenti Castelli and C. Innocenti, "Singularities, Configurations, and Displacement Functions for Manipulators," *International Journal of Robotics Research*, 5(2):52-65, 1986.
- [Leou89] J-J Leou and W-H Tsai, "Optimal Sensor Arrangement for Robot Operation Monitoring by Machine Vision," *International Journal of Robotics and Automation*, 4(2):81-91, 1989.
- [Moori88] B.W. Mooring and T.J. Pack, "Calibration Procedures for an Industrial Robot," in *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, Philadelphia, Pennsylvania, 1988, pp. 786-791.
- [Moori84] B.W. Mooring and G.R. Tang, "An Improved Method for Identifying the Kinematic Parameters in a Six-Axis Robot," in *Proceedings ASME International Computers in Engineering Conference and Exhibit*, 1984, pp. 79-84.
- [Nitza88] D. Nitzan, "Three Dimensional Vision Structure for Robot Applications," *IEEE Trans. PAMI*, 10(3):291-309, 1988.
- [PaiLe89] D.K. Pai and M.C. Leu, "Generic Singularities of Robot Manipulators," in *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, Scottsdale, Arizona, 1989, pp. 738-744.
- [Paul81] R.P. Paul, *Robot Manipulators: Mathematics, Programming, and Control*, MIT Press, Cambridge, Mass., 1981.
- [Piepe68] D.L. Pieper, "The Kinematics of Manipulators Under Computer Control," *Stanford Artificial Intelligence Laboratory*, Stanford University, AIM 72, 1968.

- [PoonL88] J.K. Poon and P.D. Lawrence, "Manipulator Inverse Kinematics Based on Joint Functions," in *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, Philadelphia, Pennsylvania, 1988, pp. 669-674.
- [Sengu87] S.H. Sengupta and H.S. Yang, "Intelligent Machine Vision System for Complex Industrial Tasks," in *Proceedings of the 1987 IEEE International Conference on Systems, Man, and Cybernetics*, Alexandria, Virginia, 1987, pp. 588-593.
- [Spets89] M.E. Spetsakis and J. Aloimonos, "Optimal Motion Estimation," in *Proceedings of the 1989 IEEE Workshop on Visual Motion*, Irvine, California, 1989, pp. 229-237.
- [Skaar87] S.B. Skaar, W.H. Brockman and R. Hanson, "Camera-Space Manipulation," *International Journal of Robotics Research*, 6(4):20-32, 1987.
- [Stone84] H.W. Stone, A.C. Sanderson and Neuman, "Arm Signature Identification," in *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, California, 1986, pp. 41-48.
- [TsaiL89] R.Y. Tsai and R.K. Lenz, "A New Technique for Fully Autonomous and Efficient 3D Robotics Hand/Eye Calibration," *IEEE Transactions on Robotics and Automation*, 5(3):345-358, 1989.
- [Vaish87] R.N. Vaishnav and E.B. Magrab, "A General Procedure to Evaluate Robot Positioning Errors," *International Journal of Robotics Research*, 6(1):59-74, 1987.
- [Wang87] C-C Wang, "Calibrating the Position of a Vision Sensor Mounted on a Robot Wrist," in *Proceedings of the 1987 IEEE International Conference on Systems, Man, and Cybernetics*, Alexandria, Virginia, 1987, pp. 461-465.
- [Weiss87] L. E. Weiss, A. C. Anderson and C. P. Neuman, "Dynamic Sensor-Based Control of Robots with Visual Feedback," *IEEE Journal of Robotics and Automation*, 5(3):404-417, 1987.
- [Whitn86] D.E. Whitney, C.A. Lozinski and J.M. Rourke, "Industrial Robot Forward Calibration Method and Results," *Journal Dyn. Systems, Measurement, and Control*, 108(1):1-8, 1986.
- [Whitn55] H. Whitney, "Mappings of the Plane onto the Plane," *Annals of Mathematics*, 62:374-470, 1955.
- [ŽakBr87] S.H. Žak, J.D. Brehove and M.J. Corless, "Control of Uncertain Systems with Unmodelled Actuator and Sensor Dynamics and Incomplete State Information," in *Proceedings of the 1987 IEEE International Conference on Systems, Man, and Cybernetics*, Alexandria, Virginia, 1987, pp. 222-226.

REAL-TIME VISUAL SERVOING

*Peter Allen
Billibon Yoshimi
Aleksandar Timcenko*

Department of Computer Science
Columbia University
New York, NY 10027

ABSTRACT

This paper describes a new real-time tracking algorithm in conjunction with a predictive filter to allow real-time visual servoing of a robotic arm. The system consists of two calibrated cameras that provide images to a real-time, pipelined-parallel optic-flow algorithm that can robustly compute optic-flow and calculate the 3-D position of a moving object at approximately 5 Hz rates. These 3-D positions serve as input to a predictive kinematic control algorithm that uses an $\alpha - \beta - \gamma$ filter to update the position of a robotic arm tracking the moving object. Experimental results are presented for the tracking of a moving model train in a variety of different trajectories.

1. INTRODUCTION

Tracking of objects by a vision system in real-time is an important problem. It has been addressed by researchers in a number of different fields including target tracking, surveillance, automated guidance systems, inspection, and monitoring. The focus of our work is to achieve a high level of interaction between a real-time vision system that is capable of tracking moving objects in 3-D and a robot arm that contains a dexterous hand that can be used to intercept, grasp and pick up a moving object. We are interested in exploring the interplay of

hand-eye coordination for dynamic grasping tasks such as grasping of parts on a moving conveyor system. However, the algorithms we have developed are applicable to a wider range of domains including the ones described above.

Previous efforts in the areas of motion tracking and real-time control are too numerous to exhaustively list here. We instead list some notable efforts that have inspired us or use similar approaches. Burt et al. [7] has focused on high speed feature detection and hierarchical scaling of images in order to meet the real-time demands of surveillance and other robotic applications. Related work has been reported by Lee and Wohn [18] and Wiklund and Granlund [28] who use image differencing methods to track motion. Corke, Paul and Wohn [10] report a feature based tracking method that uses special purpose hardware to drive a servo-controller of an arm-mounted camera. Goldenberg et al [11] have developed a method that uses temporal filtering with similar hardware to our own. Luo, Mullen and Wessel [19] report a real-time implementation of motion tracking in 1-D based on Horn and Schunk's method. Verghese et al. [27] report real-time, short-range visual tracking of objects using a pipelined system similar to our own. Safadi [23] uses a tracking filter similar to our own and a pyramid based vision system, but few results are reported with this system. Rao and Durrant-Whyte [22] have implemented a Kalman filter based de-centralized tracking system that tracks moving objects with multiple cameras. An earlier work of ours [2], explored hand-eye interaction with a simple vision algorithm and a SCARA robot that performed tracking in the X-Y plane using a single calibrated camera that was mounted on the arm. The performance of this system was inherently limited by the vision algorithm and the lack of a predictive component in the tracking

This work was supported in part by DARPA contract N00039-84-C-0165, NSF grants DMC-86-05065, DCI-86-08845, CCR-86-12709, IRI-86-57151, IRI-88-1319, North American Philips Laboratories, Siemens Corporation and Rockwell Inc.

filter to allow the arm control to compensate for vision processing delays. It also was unable to track in three dimensions.

This work is notable for the following reasons: First, the vision algorithm is robust since it is based upon detecting optic-flow in real-time as opposed to simple differencing or thresholding methods. Second, the system is capable of tracking objects in three dimensions in real-time using unregistered (but calibrated) cameras. Third, the robotic control system is able to accurately predict kinematic parameters for trajectory following of moving objects in real-time.

2. COMPUTING OPTIC-FLOW

One of the goals of this work has been to determine optical flow fields that measure image velocity at each pixel in the image. A variety of techniques for computing optic-flow fields have been used with varying results including matching based techniques [3,8,24], gradient based techniques [9,15,20] and spatio-temporal energy methods [1,13]. Optic-flow was chosen as the primitive upon which to base the tracking algorithm for the following reasons:

- The ability to track an object in three dimensions implies that there will be motion across the retinas (image planes) that are imaging the scene. By identifying this motion in each camera, we can begin to find the actual 3-D motion.
- The principal constraint in the imaging process is high computational speed to satisfy the update process for the robotic arm parameters. Hence, we needed to be able to compute image motion quickly and robustly. The Horn-Schunck optic flow algorithm (described below) is well suited for real-time computation on our PIPE image processing engine.
- We have developed a new framework for computing optic-flow robustly using an estimation-theoretic framework [25]. While this work does not specifically use these ideas, we have future plans to try to adapt this algorithm to such a framework.

Our method begins with an implementation of the Horn-Schunck method of computing optic-flow [14]. The underlying assumption of this method is the optic-flow constraint equation, which assumes image irradiance at time t and $t+\delta t$ will be the same:

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x, y, t). \quad (1)$$

If we expand this constraint via a Taylor series expansion, and drop second and higher order terms, we obtain the form of the constraint we need to compute normal velocity

$$I_x u + I_y v + I_t = 0 \quad (2)$$

where u and v are the velocities in image-space, and I_x , I_y , and I_t are the spatial and temporal derivatives in the image. This constraint limits the velocity field in an image to lie on a straight line in velocity space. The actual velocity cannot be determined directly from this constraint due to the aperture problem, but one can recover the component of velocity normal to this constraint line as:

$$V_n = -\frac{I_t}{\sqrt{I_x^2 + I_y^2}} \quad (3)$$

While computationally appealing, this method of determining optic-flow has some inherent problems. First, the computation is done on a pixel by pixel basis, creating a large computational demand. Second, the information on optic flow is only available in areas where the gradients defined above exist. A second, iterative process is usually employed to propagate velocities in image neighborhoods, based upon a variety of smoothness and heuristic constraints.

We have overcome the first of these problems by using the PIPE image processor [4,17]. The PIPE is a pipe-lined computer capable of processing 256x256x8 bit images at frame rate speeds, and it supports the operations necessary for optic-flow computation in a pixel-parallel method (a typical image operation such as convolution, warping, addition/subtraction of images can be done in one cycle - 1/60 second). The second problem is alleviated by our not needing to know the actual velocities in the image. What we need is the ability to locate and quantify gross image motion robustly. This rules out simple differencing methods which are too prone to noise and will make location of image movement difficult. Hence, a set of normal velocities at strong gradients is adequate for our task, precluding the smoothing step of the algorithm.

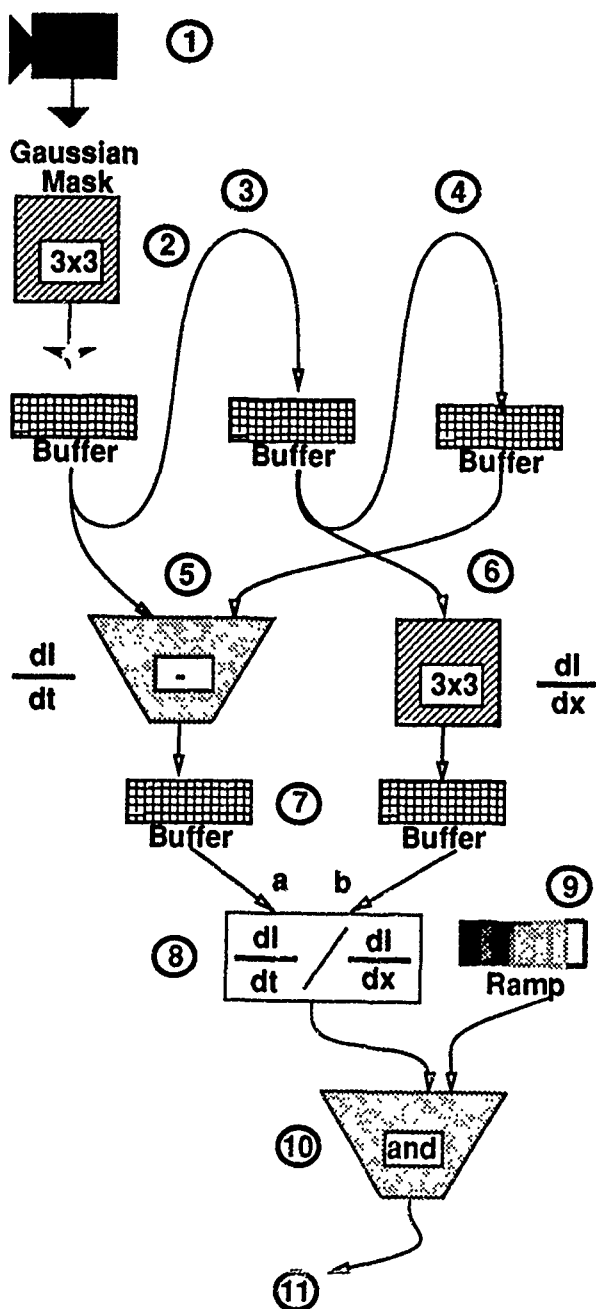


Figure 1: PIPE Motion Tracking Algorithm.

3. A REAL-TIME OPTIC-FLOW ALGORITHM

To implement our algorithm in the PIPE, we used 4 processors on the PIPE. The processors are assigned as 2 per camera - one each for the calculation of X and Y motion energy centroids in each image. We also use a special processor board (ISMAP) to perform real-time histogramming. The steps below correspond to the numbers in Figure 1 (single camera):

1. The camera images the scene and the image is sent to processing stages in the PIPE.
2. The image is smoothed by convolution with a Gaussian mask. The convolution operator is a built in operation in the PIPE and it can be performed in one frame cycle.
- 3-4. In the next 2 cycles, two more images are read in, smoothed and buffered, yielding smoothed images I_0 and I_1 and I_2 . The ability to buffer and pipeline images allows temporal operations on images, albeit at the cost of processing delays (lags) on output. There are now 3 smoothed images in the PIPE, with the oldest image lagging by 3/60 second.
5. Images I_0 and I_2 are subtracted yielding the temporal derivative I_t .
6. In parallel with step 5, Image I_1 is convolved with a 3x3 horizontal spatial gradient operator, returning the discrete form of I_x . In parallel, the vertical spatial gradient is calculated yielding I_y (not shown).
- 7-8. The results from steps 5 and 6 are held in buffers and then are input to a look-up table that divides the temporal gradient at each pixel by the absolute value of the summed horizontal and vertical spatial gradients.
- 9-10. In order to get the centroid of the motion information, we need the X and Y coordinates of the motion energy. For simplicity sake we show only the situation for the X coordinate. The gray-value ramp in Figure 1 encodes the horizontal coordinate value (0-255) for each point in the image. If we threshold the computed normal velocities, and then AND the above threshold velocities with the positional ramp, we have an image which encodes high velocity with its positional coordinates in the image. In our experiments, we thresholded all

velocities below 10 pixels per 60 msec. to zero velocity.

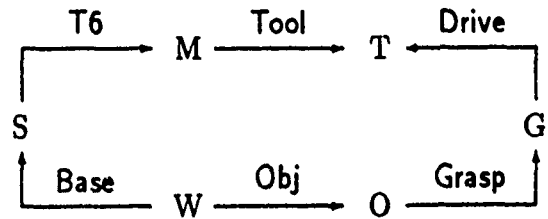
11. By taking this result and histogramming it, via a special stage of the PIPE which performs histograms at frame rate speeds, we can find the centroid of the moving object by finding the mean of the resulting histogram. Histogramming the high velocity position encoded images yields 256 16-bit values (a result for each intensity in the image). These 256 values can be read off the PIPE via a parallel interface in about 10 ms. This operation is performed in parallel to find the moving objects Y centroid (and in parallel for X and Y centroids for camera 2). The total associated delay time for finding the centroid of a moving object becomes 15 cycles or 0.25 seconds.

The same algorithm is run in parallel on the PIPE for the second camera. Once the motion centroids are known for each camera, they are back-projected into the scene using the camera calibration matrices and triangulated to find the actual 3-D location of the movement. Because of the pipelined nature of the PIPE, a new X or Y coordinate is produced every 1/60 second with this delay.

The system exhibits an interesting mix of local and global computations, separated by processors and update rates. The PIPE is able to perform the local optic-flow computation at video rates (but with delay), the ISMAP board gathers global histogram statistics, and these are then shipped via a high-speed interface to a host where the stereo triangulation and kinematic control algorithms reside, updating the arm parameters every 30 msec.

4. ROBOTIC ARM CONTROL

The updated information from the cameras is used to predict the next set point for the robotic arm that is tracking the moving object. We are using RCCL [12] to control the robotic arm (a PUMA 560). RCCL (Robot Control C Language) allows the use of C programming constructs to control the robot as well as defining transformation equations (as described in [21]). The transformation equations permit dynamic updating of arm position by generating the 4x4 transform of the moving object's position from the vision system and sending this information to the arm control algorithm (see Figure 2).



W is world-coordinate frame

S is robot shoulder coordinate frame

M is 6th joint coordinate frame

T is tool (gripper) coordinate frame

G is grasping position coordinate frame

O is moving object coordinate frame

Base is constant transform between *W* and *S*

T6 is variable transform computed by RCCL in each sampling interval

Tool is variable transform defined by Utah-MIT hand kinematics

Drive is the transform introduced internally by RCCL to obtain straight-line motion in Cartesian coordinates

Grasp is constant transform which defines grasping point relative to the moving object

Obj is variable transform defined by vision subsystem outputs - it defines the position of the moving object in the world coordinate frame

Figure 2: Transform Equation. Graph nodes represent coordinate frames and graph edges represent 4x4 coordinate transforms.

5. PREDICTIVE FILTERING

The 3-D values of the moving object's position determined by the vision system are both noisy and out of date due to the processing delays. To alleviate this, the system has been modeled as a standard second order system, and the input to the system is a series of filtered predictions. In choosing a predictive filter for tracking, the

major constraints are knowledge of the nature of the motion to be tracked, knowledge of the noise characteristics of the sensor, and computational cost of the filter. Kalman filtering generates time-variable tracking coefficients that are determined by *a priori* models for the statistics of measurement noise and target dynamics. Because of its dynamic nature, it imposes a computational cost which can be difficult to incur in certain real-time applications. Fixed-coefficient filters have the advantage of simple implementation using fixed parameters for the filter gains [6]. In a few particularly simple target tracking problems, it is possible to derive closed-form steady-state solutions for the associated Kalman filter covariance matrix and the corresponding filter gains. Such solutions can be used to avoid real-time computation of the complete filter equations [16], and can approach the Kalman filter in steady-state performance.

We have implemented an $\alpha - \beta - \gamma$ filter [5,23] which includes an acceleration estimate as well as a position and velocity estimate. Below are the prediction equations for the system (4-6) and the correction equations (7-9) using the filter parameters $\alpha - \beta - \gamma$:

$$x_{k+1|k} = x_{k|k} + T v_{k|k} + \frac{1}{2} T^2 a_{k|k} \quad (4)$$

$$v_{k+1|k} = v_{k|k} + T a_{k|k} \quad (5)$$

$$a_{k+1|k} = a_{k|k} \quad (6)$$

$$x_{k+1|k+1} = x_{k+1|k} + \alpha_{k+1} [z_{k+1} - x_{k+1|k}] \quad (7)$$

$$v_{k+1|k+1} = v_{k+1|k} + \frac{1}{T} \beta_{k+1} [z_{k+1} - x_{k+1|k}] \quad (8)$$

$$a_{k+1|k+1} = a_{k|k} + \frac{1}{2T^2} \gamma_{k+1} [z_{k+1} - x_{k+1|k}] \quad (9)$$

$x_{k|k}$ is the position estimate at time interval k

$x_{k+1|k}$ is the position estimate at time interval $k+1$ given k samples

$v_{k|k}$ is the velocity estimate at time interval k

$a_{k|k}$ is the acceleration estimate at time interval k

α_k is the position tracking parameter at k

β_k is the velocity tracking parameter at k

γ_k is the acceleration tracking parameter at k

T is the sampling period

z_k is measured position at time k

The gains for the fixed coefficient filters represent a compromise between noise reduction and maneuver-following capability in the steady-state. Kalata [16] has shown that the optimal steady-state gains for a $\alpha - \beta - \gamma$ filter are given by:

$$\frac{\gamma^2}{4(1-\alpha)} = \lambda^2 \quad (10)$$

$$\beta = 2(2-\alpha) - 4\sqrt{1-\alpha} \equiv \alpha = \sqrt{2\beta} - \frac{1}{2}\beta \quad (11)$$

$$\gamma = \frac{\beta^2}{\alpha} \quad (12)$$

$$\lambda = \frac{T^2 \sigma_a}{\sigma_n} \quad (13)$$

where T is the sample period, σ_a is the position uncertainty due to variance of the acceleration, and σ_n is measurement noise variance. In practice, it is very difficult to accurately estimate the variances above. We treat this ratio of variances as a free parameter in calculating the filter gains, and have found a suitable value empirically.

6. EXPERIMENTAL RESULTS

The system shown in Figure 3 was used to test the algorithm. Two CCD cameras with 25mm lenses were mounted approximately 1.5 meters apart, with the moving object's full trajectory in the field of view. The cameras were calibrated by placing an LED in a number of known positions along the trajectory determined by the world coordinate system of the robot arm. Each camera imaged the LED and computed its centroid, from which a least-squares calibration matrix was calculated [26]. The best calibration results are obtained with the cameras orthogonal to each other. In calibration situations like this we were able to determine the three-dimensional position of an LED to within 10 millimeters in X, Y and Z coordinates. The experimental results were obtained with the cameras mounted approximately 15 cm above the plane of the table pointing down at an angle of about 10 degrees from horizontal.

In the experiments, a model train was tracked by vision as it moved around a circular track and an oval track and the arm was commanded to follow each trajec-

tory. The train was moving at a velocity of approximately 25 cm/sec. The results shown below were obtained without the robotic hand mounted on the system. Figure 4 shows the left and right camera images of the moving train, and Figure 5 shows the thresholded motion-energy for each camera found by the algorithm. Figure 6 is a plot of the camera triangulated and predicted X and Y trajectories of the train. Figure 7 is the actual filtered arm trajectory obtained by following the train as it moves on an oval track. The trajectory is quite close to the actual trajectory except as the camera distance to the moving object increases. The algorithm is less accurate here for two reasons: first, the inherent stereo digitization error increases with distance from the cameras, and secondly, the profile of the moving object is quite different at this position, yielding two views whose motion centroids may not correspond to the same point in space, thus inducing triangulation error. Figure 8 is the actual filtered trajectory of the arm following the train as it moves on a circular track. As in the oval track, the trajectory is stable and repeatable, but also suffers from the two stereo induced problems mentioned above. The results are also available on video tape.

7. FUTURE WORK AND CONCLUSIONS

The algorithm described here has worked quite well in providing real-time visual servoing of a robotic arm, a fairly difficult task. We hope to continue to improve this algorithm, specifically by reducing the stereo error problems discussed above. One approach is to perform a better calibration of the camera system, and the other is to try to better isolate centroids in each separate camera to be the same physical point. Various heuristics are being investigated on this front.

We also are interested in using a Kalman filter approach with dynamic gains for prediction. There is an added computational cost this approach, but it may yield more accurate trajectories, particularly with more complex trajectories. In addition, we are exploring tracking multiple objects using the PIPE's ability to work on a region by region basis.

Finally, we are working on using the attached robotic hand to intercept and pick up the moving objects. An interesting approach we are currently exploring is to use vision to servo the hand and object together.

References

1. Adelson, E. H. and J. R. Bergen, "Spatio-temporal energy models for the perception of motion," *Journal of the Optical Society of America*, vol. 2, no. 2, pp. 284-299, 1985.
2. Allen, Peter, "Real-time motion tracking using spatio-temporal filters," *Proceedings of DARPA Image Understanding Workshop*, Morgan-Kaufman Publishers, Palo Alto, May 1989.
3. Anadan, P., *Measuring visual motion from image sequences*, Technical Report COINS-TR-87-21, COINS Department, University of Massachusetts, Amherst, 1987.
4. Aspex., *PIPE User's Manual*, Aspex Incorporated, 530 Spring St., New York, NY.
5. Bar-Shalom, Y. and T. Fortmann, *Tracking and data association*, Academic Press, 1988.
6. Blackman, Samuel, *Multiple-target tracking with radar applications*, Artech House, Dedham, MA, 1986.
7. Burt, P. J., J. R. Bergen, R. Hingorani, R. Kolczynski, W. A. Lee, A. Leung, J. Lubin, and H. Shvayster, "Object tracking with a moving camera," *IEEE Workshop on Visual Motion*, pp. 2-12, Irvine, CA, March 20-22, 1988.
8. Burt, P. J., C. Yen, and X. Xu, "Multi-resolution flow-through motion analysis," *Proceedings of the IEEE CVPR Conference*, pp. 246-252, 1983.
9. Buxton, B. F. and H. Buxton, "Computation of optic flow from the motion of edge features in image sequences," *Image and Vision Computing*, no. 2, 1984.
10. Corke, Peter, Richard Paul, and K. Wohn, *Video-rate visual servoing for sensory-based robotics*, Technical Report, Grasp Laboratory, Department of Computer and Information Science, University of Pennsylvania, Philadelphia.
11. Goldenberg, Richard, Wan Chi Lau, Alfred She, and Alan Waxman, "Progress on the prototype PIPE," *IEEE Conference on Robotics and Automation*, Raleigh, N. C., March 31 - April 3, 1987.
12. Hayward, Vincent and Richard Paul, "Robot manipulator control under UNIX," *Proc. of the 13th ISIR*, pp. 20:32-20:44, Chicago, April 17-21, 1983.

13. Heeger, David, "A model for extraction of image flow," *First International Conference on Computer Vision*, London, 1987.
14. Horn, B. K. P., *Robot vision*, M.I.T. Press, 1986.
15. Horn, B. K. P. and B. Schunk, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185-203, 1983.
16. Kalata, Paul, "The tracking index: A generalized parameter for $\alpha - \beta$ and $\alpha - \beta - \gamma$ trackers," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-20, no. 2, pp. 174-182, March 1984.
17. Kent, E. W., M. O. Shneier, and R. Lumia, "PIPE: Pipelined image processing engine," *Journal of Parallel and Distributed Computing*, no. 2, pp. 50-78, 1985.
18. Lee, Sang Wook and K. Wohn, *Tracking moving objects by a mobile camera*, Technical Report MS-CIS-88-97, University of Pennsylvania, Department of Computer and Information Science, Philadelphia, November 1988.
19. Luo, Ren C., Robert E. Mullen Jr., and Daniel E. Wessell, "An adaptive robotic tracking system using optical flow," *IEEE Conference on Robotics and Automation*, pp. 568-573, Philadelphia, 1988.
20. Nagel, H. H., "On the estimation of dense displacement vector fields from image sequences," *Workshop on motion: Representation and Perception*, pp. 59-65, Toronto, 1983.
21. Paul, Richard, *Robot Manipulators*, MIT Press, Cambridge, MA, 1981.
22. Rao, B. S. Y. and H. F. Durrant-Whyte, "A fully decentralized algorithm for multi-sensor kalman filtering," Report OUEL 1787/89, Dept. of Engineering Science, University of Oxford.
23. Safadi, Reem Bassam, "An adaptive algorithm for robotics and computer vision application," Tech. Report MS-CIS-88-05, Department of Computer and Information Science, University of Pennsylvania, January 1988.
24. Scott, G. L., "Four-line method of locally estimating optic flow," *Image and Vision Computing*, 5(2), 1986.
25. Singh, Ajit, "Image flow computation: an estimation theoretic framework, unification and integration," Ph.D. Dissertation, Department of Computer Science, Columbia University, New York, New York, May 1990.
26. Sutherland, I. E., "Three dimensional input by tablet," *Proceedings of IEEE*, vol. 62, no. 4, pp. 453-461, April 1974.
27. Verghese, Gilbert, Karey Gale Lynch, and Charles R. Dyer, "Real-time motion tracking of three-dimensional objects," *IEEE International Conference on Robotics and Automation*, Cincinnati, May 13-18, 1990.
28. Wiklund, Johan and Gosta Granlund, "Tracking of multiple moving objects," in *Time Varying Image Processing and Moving Object Recognition*, ed. V. Cappellini, pp. 241-249, Elsevier Science Publishers, 1987.

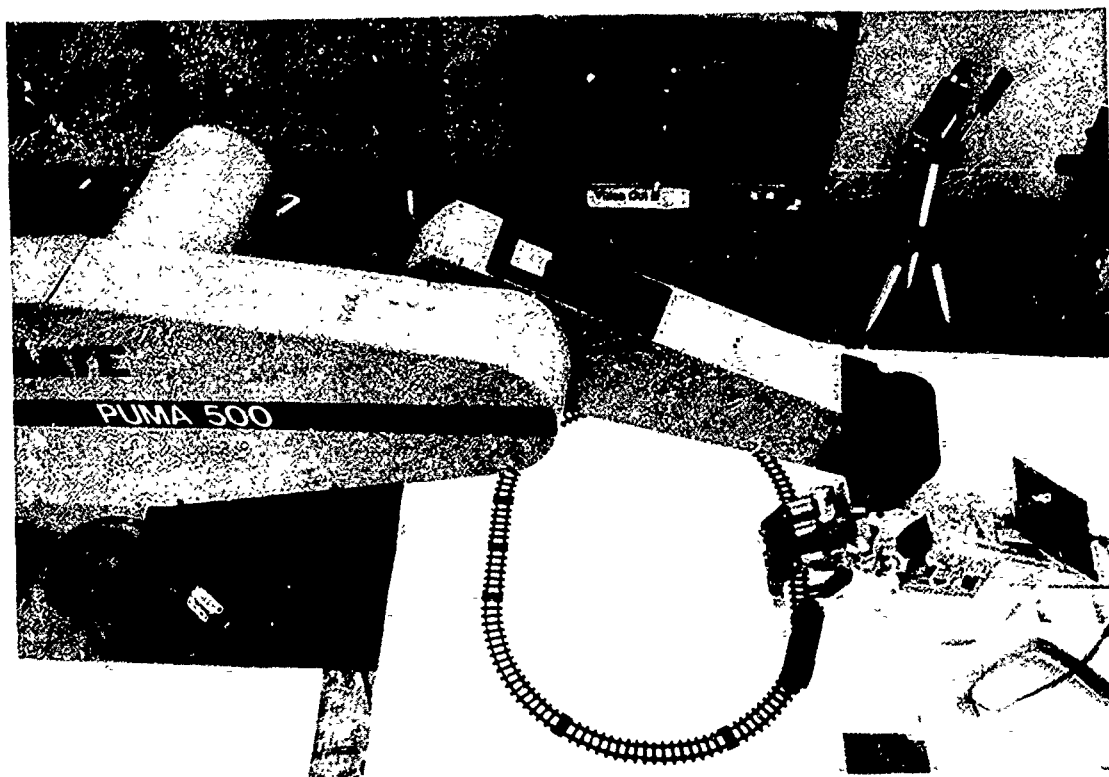


Figure 3. Experimental Hardware.

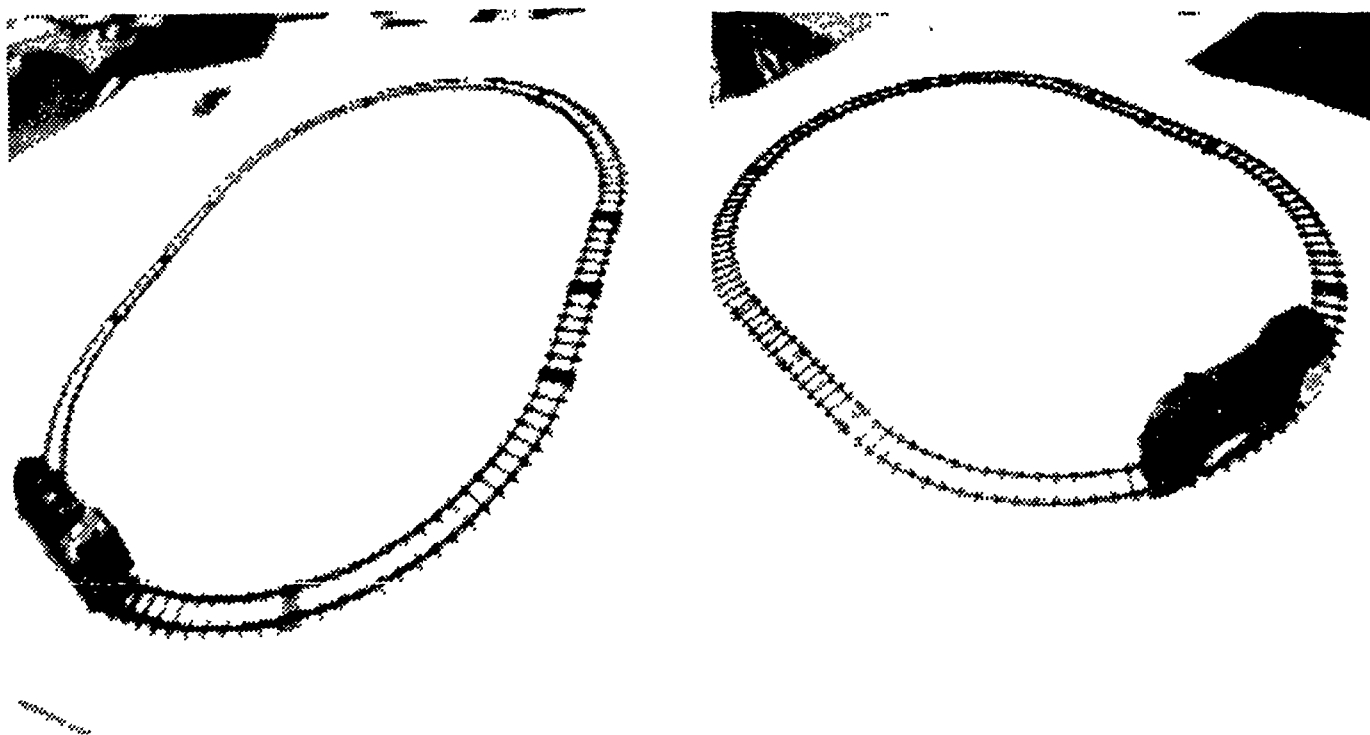
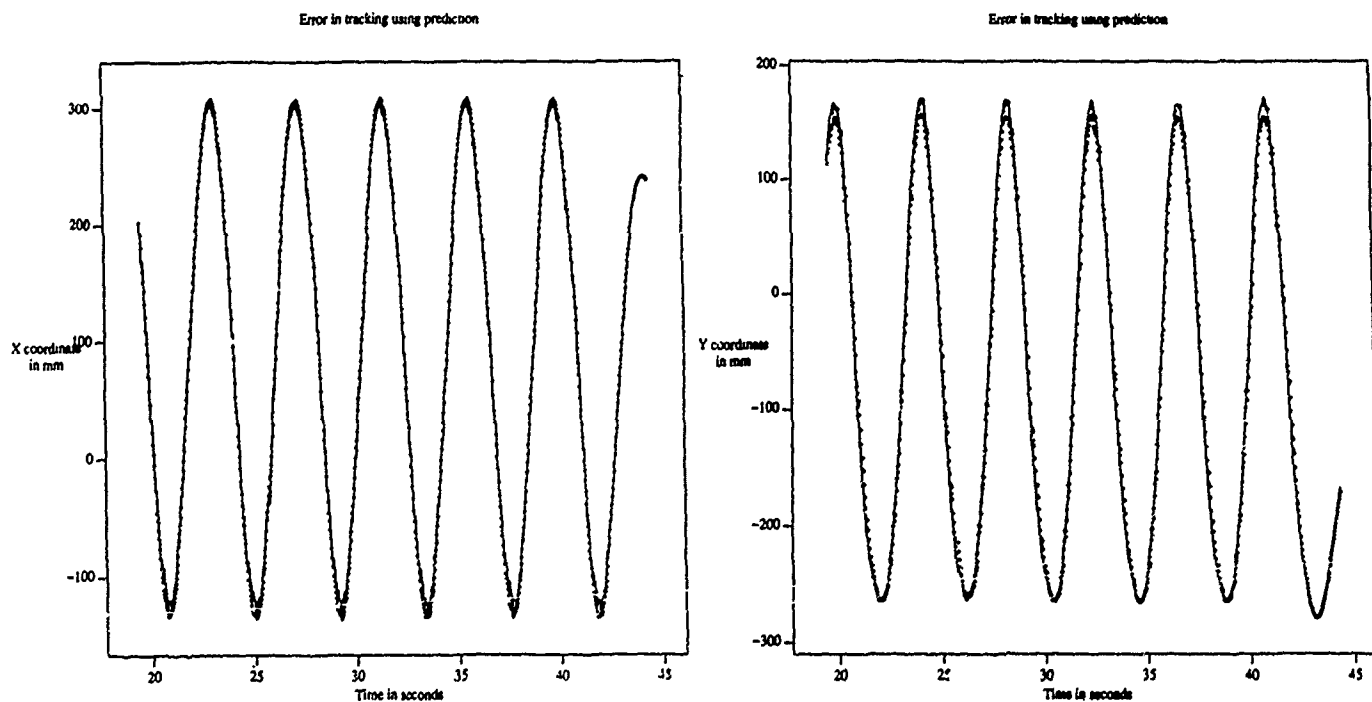


Figure 4. Left and Right camera images



**Figure 5. Motion Energy derived from Optic-Flow
(left and right cameras).**



**Figure 6. X and Y triangulated (bullets) and predicted
trajectories (straight line), circular path**

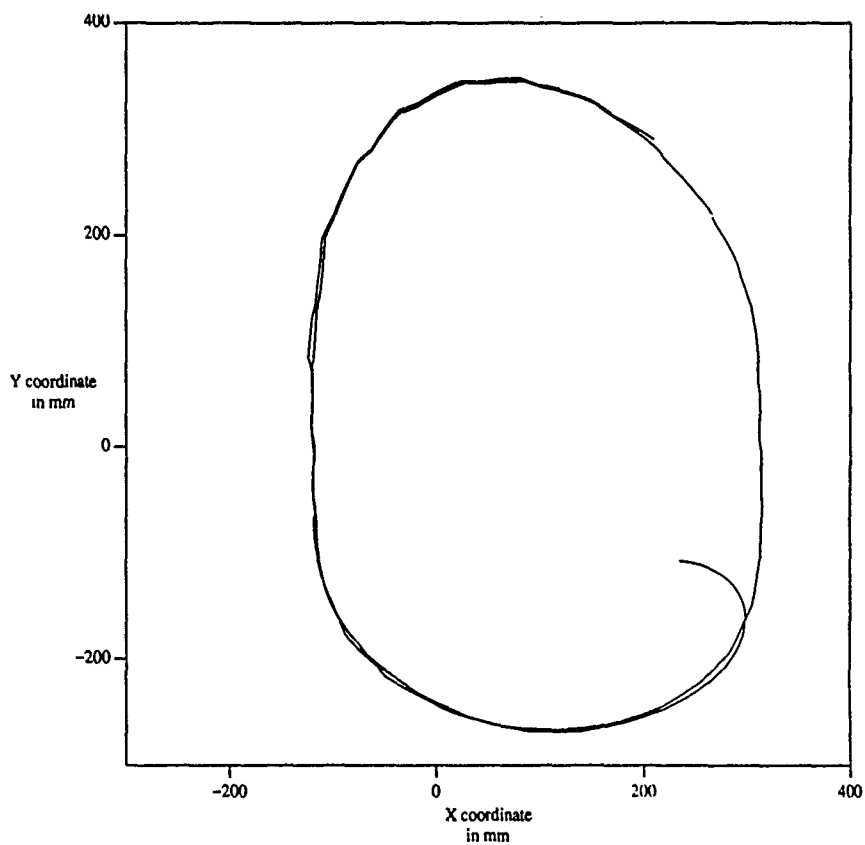


Figure 7. Tracked arm path, oval trajectory.

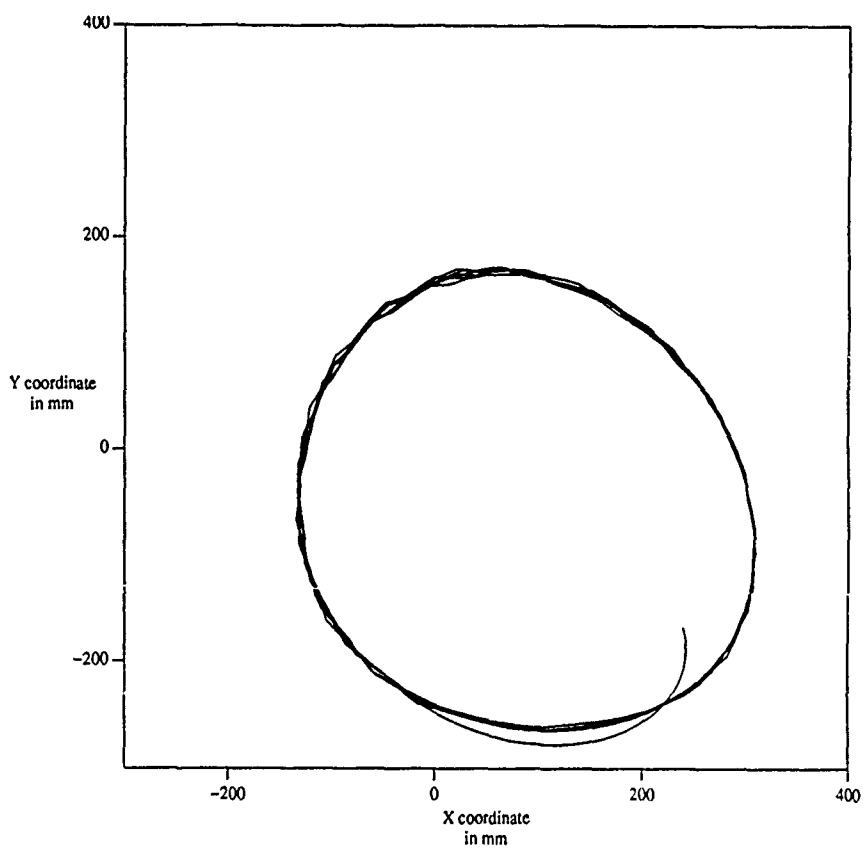


Figure 8. Tracked arm path, circular trajectory.

Force/Torque Sensing for Object Recognition

William A. Wolovich

Division of Engineering,
Brown University,
Providence, RI 02912

Abstract

Our primary objective, with respect to force/torque sensing, is to determine how touch can be used for object recognition via force-contact motion control along the surface of an unknown object. Since force information employs only "local" information, in that it involves only the contacted object, and not the environment, far less information need be processed (when compared to vision) in order to identify a contacted object. We will present a new technique for moving the end-effector of a robot along a surface, without explicit knowledge of the surface, in order to obtain trajectory information which can subsequently be used to identify the object.

1 Overall Approach

The approach that has been taken to achieve this objective might be classified as *continuous contact force/torque sensing*. Unlike isolated point sensing, which is slow and provides only sparse data, this sensing is relatively fast (assuming that surface contact is maintained). Moreover, this approach provides a continuum of data points which correspond to an entire trajectory along the surface.

The key to much of our work is a modified application of a relatively new form of robot control, termed *dual-drive*, which enables a robot to track along an unknown surface at a specified speed and direction using only positional and force/torque feedback. Dual-drive control represents a new form of robotic "hybrid" control. The basic idea behind this form of control is to directly drive each link axis by an "appropriate combination" of two different Cartesian error signals, one a velocity and the other a force. The velocity error signal acts to increase (decrease) each link rate whenever the Cartesian velocity is too low (high). The force error signal acts to increase (decrease) the torque applied by each link actuator to the end effector whenever the Cartesian force experienced by the end effector is too low (high). It is of interest to note that this form of control can also be used in the case of mobile robots, where the task can be stated as that of moving the entire robot in a direction which is tangential to a sensed surface (similar to end-effector motion

along an unknown surface). Future research plans will fully exploit this potential in the case of mobile robots.

2 Prior Results

Dual-drive control already had been developed and successfully tested for robot crank-turning experiments, as well as smooth surface tracking, prior to the initiation of this grant¹. [Kazanzides, *et al.*, 1989, Bradley and Wolovich, 1990]. Furthermore, a considerable amount of time and energy has been expended to develop SIERA, *System for Implementing and Evaluating Robotic Algorithms*, a hybrid multiprocessor system which represents a key component in the physical implementation of dual-drive control [Kazanzides, *et al.*, 1988]. SIERA also has been used to study and evaluate the consequences of automatically changing control laws as required; e.g. when surface contact is made or lost, or when additional sensory information becomes available. The IBM 7565 robot in our Laboratory for Engineering Man/Machine Systems (LEMS) at Brown University (controlled by SIERA) will continue to represent the primary "test-bed" for virtually all of our non-mobile robot experiments.

3 Current Status

Our most recent work has focused on developing procedures for storing and interpreting trajectory information obtained via the strain gauge (force/torque) sensors mounted on the end-effector of the IBM 7565 robot, as it moves along and around a variety of different 2-D surfaces using the dual-drive algorithm. Before any trajectory information can be gathered, an object must first be found. In [Bradley, 1990], a technique termed "structured wandering" is employed to achieve this initial objective. More specifically, the search space is assumed to be known in size, thereby limiting the prescribed environment. We also assume that there are one or more unknown objects in this space. This space is then partitioned into N rows, which subsequently define the search path, as depicted in Figure 1. Each row, and the two columns, are followed by force complying in the desired direction. In this way, the robot moves along the search path using a rather simple, position² control strategy,

¹Future investigations will be supported, in part, by the NSF and DARPA under Grant No. IRI-8905436

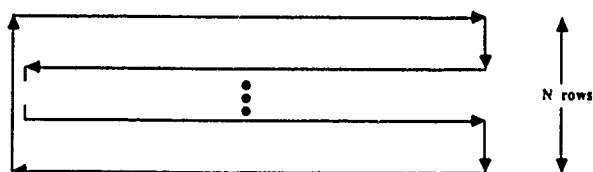


Figure 1: Sample Search Path

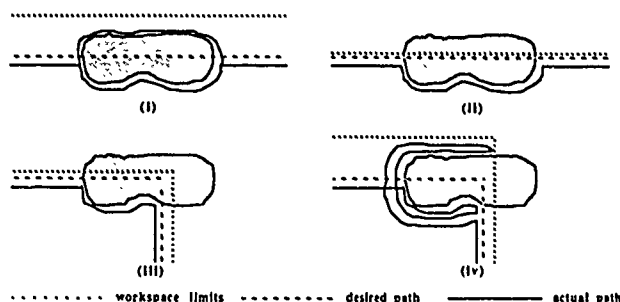


Figure 2: Tracing an Object

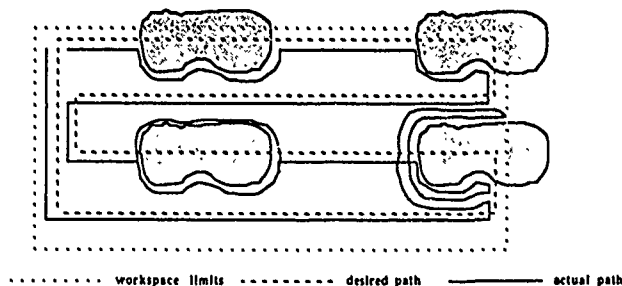


Figure 3: Object Tracing Limits

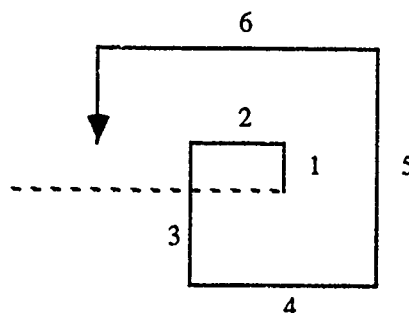


Figure 4: Object Searching

until an object is detected. Once an object is encountered (by force detection), control is "switched" from positional to dual-drive and the robot begins to trace the 2-D surface of the object.

Once surface contact has been established, any one of the following can occur (depending on when the desired path, robot joint limits and the object intersect). Figure 2 illustrates each possibility.

(i) If the object lies completely inside the work space, the object's surface will be followed one complete time, plus the fraction of time necessary to intersect the opposite side of the object where the unblocked path would intersect the surface. The robot then continues along the N row path.

(ii) If the object lies partly outside of the robot's limits, then the robot can follow only part of the surface without reaching a limit. The rest of the surface either must be estimated or ignored.

(iii) When the robot reaches a corner of the workspace, the least amount of data is available since most of the object is beyond the robot's limits. Again, the remainder of the surface must be estimated or ignored, especially if only the immediate environment is of interest.

(iv) The most complicated situation occurs when the desired path splits the object in such a way that it must be followed multiple times to obtain the maximum amount of data possible. Another possibility is to follow the object as if it were at a corner of the workspace and to then estimate the remainder. Figure 3 illustrates how all four possibilities could occur in a single search.

In certain situations, surface contact may be lost. One of the most important operations that must occur, in order to re-establish contact, is a continuous change in real-

time control mode between the dual-drive mode (which is used when contact has been established) and the *fcomply* mode (which is used to search for the surface once contact is lost). If the dual-drive mode is operative and contact is lost during surface following, control is automatically "switched" to the *fcomply* mode while the robot searches for the surface. Figure 4 depicts the general path of such a search when the robot has been following an object in a counter-clockwise direction. In the clockwise case, the mirror image of the depicted search path is implemented.

Figure 5 illustrates three situations when the robot might lose contact with a surface. In case (a), contact may be lost because either a force error is over-compensated for, or the specified velocity is too large relative to the force being used to maintain surface contact. Case (b) illustrates a more likely situation where contact might be lost as the result of an abrupt angular change ($< 90^\circ$), while surface contact will almost surely be lost in case (c). Once surface contact has been re-established, control is then automatically switched back to the dual-drive mode.

Once the complete surface of an object has been traced, and the data collected, the object can be identified. The complexity of this task will, of course, depend on our knowledge of the environment, including the assumptions made relative to the number, shape and type of objects that might be encountered. The experiments conducted thus far have assumed the presence of only simple polygons and ellipses, thus simplifying their identification from surface tracking data.

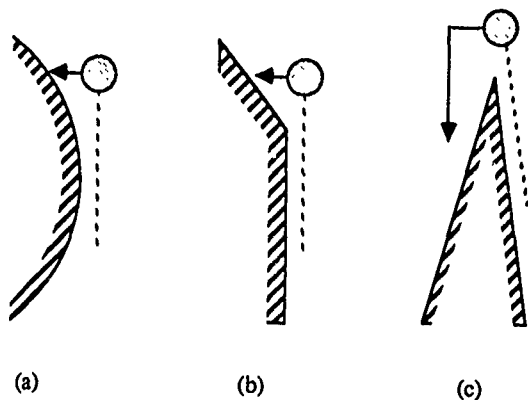


Figure 5: Losing Surface Contact

One such experiment employed three simple objects (a four sided polygon, an ellipse and a triangle) placed arbitrarily, but completely, within the workspace of the robot. Figure 6 shows the actual path that the robot followed during the experiment. The contours of the objects are easily distinguishable from the overall path. The "glitches" that appear at certain of the vertices of the objects are, in fact, the path followed (via the fcomply control mode) when surface contact was temporarily lost. Although the ellipse is followed twice, as is evident from the actual path obtained in the experiment, only data from the the initial tracking is depicted in (b) and (c) of the figure. Figure 6(b) shows the actual data collected for each of the objects. The short "gaps" which appear in the data are due to the fact that the data collecting frequency was only 2 Hz. As a consequence, the gaps correspond to the "distance" between two consecutive pieces of data. Since the object identification scheme employed assumes a connection between consecutive data points, this does not represent a problem from the point of view of object recognition. Finally, Figure 6(c) displays the actual data of each object with the identifier estimates superimposed. Note that the two are virtually indistinguishable. More detailed information about this experiment is contained in [Kawamura, 1990].

4 Conclusions and Future Plans

Our research, thus far, has proven the effectiveness of force/torque sensing for surface tracking for simple object identification. Proposed additional experiments will employ the IBM 7565 manipulator in order to extend the results obtained, thus far, in a number of different directions. More specifically, we plan to incorporate some of the algorithms developed by G. Taubin [1989] here in LEMS, in order to recognize more complex 2-D objects using force/torque sensing. We have already begun to extend our results to the 3-D case in order to first recognize simple 3-D objects, such as cubes, cones and balls, with the goal of eventually recognizing more complex objects, which may not have such simple shapes. Here again, the algorithms developed by G. Taubin should

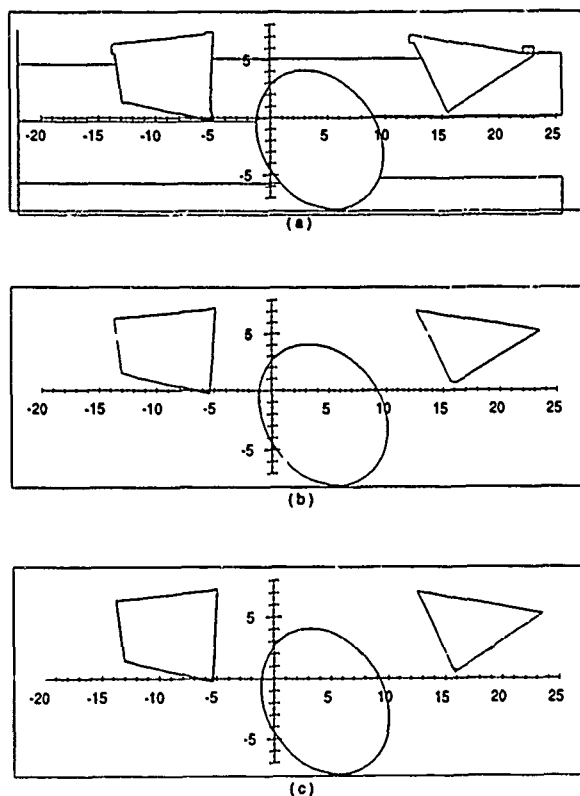


Figure 6: Experimental Data

play an important role in our investigations. We also plan to integrate vision information in the very near future, first to enhance object acquisition, and later to complement the object recognition task via *sensor fusion*; i.e. simultaneously utilizing both vision and touch information in order to obtain an "optimal" identifier.

5 References

- [Kazanzides, et al., 1989] P. Kazanzides, N. S. Bradley and W. A. Wolovich, *Dual-Drive Force/Velocity Control: Implementation and Experimental Results*, 1989 IEEE Conference on Robotics and Automation, Scottsdale, Arizona, May 14-19, 1989.
- [Kazanzides, et al., 1988] P. Kazanzides, H. Wasti and W. A. Wolovich, *SIERA: A Multiprocessor System for Real-Time Robotic Control*, Journal of Information Sciences, Vol. 44 (3), pp 225-247, April, 1988.
- [Bradley, 1990] N. S. Bradley, *Force Control Methods for Environment Interaction and Control*, Sc.M. Thesis, Division of Engineering, Brown University, May, 1990.
- [Bradley and Wolovich, 1990] N. S. Bradley and W. A. Wolovich, *Task Formation Using Multiple Force Controllers*, Technical Report LEMS-72, Brown University, June, 1990.
- [Kawamura, 1990] K. Kawamura, *Implementation of Force Control Methods for Object Recognition*, Sc.M. Thesis, Division of Engineering, Brown University, May, 1990.
- [Taubin, 1989] G. Taubin, *About Shape Descriptors and Shape Matching*, Technical Report LEMS-57, Brown University, March, 1989.